User's Manual

CubeSuite+ V2.01.00

Integrated Development Environment

User's Manual: RH850 Build

Target Device
RH850 Family

NOTICE:
There are corrections on page 28, page 29, page 86 and page 96 in this document.

All information contained in these matenals, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Electronics Corp. website (http://www.renesas.com).

Renesas Electronics
www.renesas.com

Rev.1.00 Apr 2013

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(2012.4)

How to Use This Manual

This manual describes the role of the CubeSuite+ integrated development environment for developing applications and systems for RH850 family, and provides an outline of its features.

CubeSuite+ is an integrated development environment (IDE) for RH850 family, integrating the necessary tools for the development phase of software (e.g. design, implementation, and debugging) into a single platform.

By providing an integrated environment, it is possible to perform all development using just this product, without the need to use many different tools separately.

Readers

This manual is intended for users who wish to understand the functions of the CubeSuite+ and design software and hardware application systems.

Purpose

This manual is intended to give users an understanding of the functions of the CubeSuite+ to use for reference in developing the hardware or software of systems using these devices.

Organization

This manual can be broadly divided into the following units.

CHAPTER 1 GENERAL
CHAPTER 2 FUNCTIONS
CHAPTER 3 BUILD OUTPUT LISTS
APPENDIX A WINDOW REFERENCE
APPENDIX B COMMAND REFERENCE
APPENDIX C INDEX

How to Read This Manual

It is assumed that the readers of this manual have general knowledge of electricity, logic circuits, and microcontrollers.

Conventions

Data significance:	Higher digits on the left and lower digits on the right
Active low representation:	$\overline{\text{XXX}}$ (overscore over pin or signal name)
Note:	Footnote for item marked with Note in the text
Caution:	Information requiring particular attention
Remark:	Supplementary information
Numeric representation:	Decimal ... XXXX
	Hexadecimal ... 0xXXXX

Related Documents

The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

Document Name		Document No.
CubeSuite+ Integrated Development Environment User's Manual	Start	R20UT2444E
	V850 Design	R20UT2134E
	R8C Design	R20UT2135E
	RL78 Design	R20UT2136E
	78K0R Design	R20UT2137E
	78K0 Design	R20UT2138E
	RH850 Coding	R20UT2584E
	RX Coding	R20UT2470E
	V850 Coding	R20UT0553E
	Coding for CX Compiler	R20UT2139E
	R8C Coding	R20UT0576E
	RL78, 78K0R Coding	R20UT2140E
	78K0 Coding	R20UT2141E
	RH850 Build	This manual
	RX Build	R20UT2472E
	V850 Build	R20UT0557E
	Build for CX Compiler	R20UT2142E
	R8C Build	R20UT0575E
	RL78, 78K0R Build	R20UT2143E
	78K0 Build	R20UT0783E
	RX Debug	R20UT2350E
	V850 Debug	R20UT2446E
	R8C Debug	R20UT0770E
	RL78 Debug	R20UT2445E
	78K0R Debug	R20UT0732E
	78K0 Debug	R20UT0731E
	Analysis	R20UT2447E
	Message	R20UT2448E

Caution The related documents listed above are subject to change without notice. Be sure to use the latest edition of each document when designing.

All trademarks or registered trademarks in this document are the property of their respective owners.

TABLE OF CONTENTS

CHAPTER 1 GENERAL ... 8

1.1 Overview ... 8
1.2 Copyrights ... 9
1.3 Features ... 10

CHAPTER 2 FUNCTIONS ... 11

2.1 Overview ... 11
2.1.1 Create a load module ... 11
2.1.2 Create a user library ... 12
2.2 Change the Build Tool Version ... 13
2.3 Set Build Target Files ... 14
2.3.1 Add a file to a project ... 14
2.3.2 Remove a file from a project ... 18
2.3.3 Remove a file from the build target ... 18
2.3.4 Classify a file into a category ... 19
2.3.5 Change the file display order ... 20
2.3.6 Update file dependencies ... 20
2.4 Set the Type of the Output File ... 24
2.4.1 Change the output file name ... 24
2.4.2 Output an assemble list ... 27
2.4.3 Output map information ... 27
2.4.4 Output library information ... 28
2.5 Set Compile Options ... 30
2.5.1 Perform optimization with the code size precedence ... 31
2.5.2 Perform optimization with the execution speed precedence ... 31
2.5.3 Add an include path ... 31
2.5.4 Set a macro definition ... 33
2.5.5 Change the register mode ... 34
2.6 Set Assemble Options ... 35
2.6.1 Add an include path ... 35
2.6.2 Set a macro definition ... 37
2.7 Set Link Options ... 39
2.7.1 Add a user library ... 40
2.7.2 Prepare for using the overlaid section selection function ... 41
2.8 Set Hex Output Options ... 50
2.8.1 Set the output of a hex file ... 50
2.8.2 Fill the vacant area ... 52
2.8.3 Combine multiple objects ... 54
2.9 Set Create Library Options ... 55
2.9.1 Set the output of a library file ... 55
2.10 Set Build Options Separately ... 57

2.10.1 Set build options at the project level ... 57
2.10.2 Set build options at the file level ... 57
2.11 Make Settings for Build Operations ... 61
2.11.1 Import the build options of other project ... 61
2.11.2 Set the link order of files ... 62
2.11.3 Change the build order of subprojects ... 65
2.11.4 Display a list of build options ... 66
2.11.5 Change the build target project ... 66
2.11.6 Add a build mode ... 68
2.11.7 Change the build mode ... 69
2.11.8 Delete a build mode ... 71
2.11.9 Set the current build options as the standard for the project ... 72
2.12 Run a Build ... 73
2.12.1 Run a build of updated files ... 75
2.12.2 Run a build of all files ... 75
2.12.3 Run a build in parallel with other operations ... 76
2.12.4 Run builds in batch with build modes ... 77
2.12.5 Compile/assemble/link multiple files simultaneously ... 78
2.12.6 Compile/assemble individual files ... 79
2.12.7 Stop running a build ... 80
2.12.8 Save the build results to a file ... 80
2.12.9 Delete intermediate files and generated files ... 81
2.13 Estimate the Stack Capacity ... 82

CHAPTER 3 BUILD OUTPUT LISTS ... 83

3.1 Assemble List File ... 83
3.1.1 Structure of the assemble list ... 83
3.1.2 Assemble list ... 84
3.1.3 Section list ... 85
3.1.4 Command line information ... 85
3.2 Link Map File ... 86
3.2.1 Structure of link map ... 86
3.2.2 Option information ... 87
3.2.3 Error information ... 87
3.2.4 Link map information ... 88
3.2.5 Total section size ... 88
3.2.6 Symbol information ... 89
3.2.7 Cross reference information ... 90
3.3 Link Map File (When Objects Are Combined) ... 92
3.3.1 Structure of link map ... 92
3.3.2 Header information ... 93
3.3.3 Option information ... 93
3.3.4 Error information ... 94
3.3.5 Entry information ... 94
3.3.6 Combined address information ... 94
3.3.7 Address overlap information ... 95
3.4 Library List File ... 96
3.4.1 Structure of the library list ... 96

3.4.2 Option information ... 96
3.4.3 Error information ... 97
3.4.4 Library information ... 97
3.4.5 Module, section, and symbol information within the library ... 98
3.5 Intel HEX File ... 100
3.5.1 Structure of the Intel HEX file ... 100
3.5.2 Start linear address record ... 102
3.5.3 Expanded linear address record ... 102
3.5.4 Start address record ... 102
3.5.5 Expanded address record ... 103
3.5.6 Data record ... 103
3.5.7 End record ... 104
3.6 Motorola S-record File ... 105
3.6.1 Structure of the Motorola S-record file ... 105
3.6.2 S0 record ... 106
3.6.3 S1 record ... 107
3.6.4 S2 record ... 107
3.6.5 S3 record ... 107
3.6.6 S7 record ... 108
3.6.7 S8 record ... 108
3.6.8 S9 record ... 109

APPENDIX A WINDOW REFERENCE ... 110

A.1 Description ... 110

APPENDIX B COMMAND REFERENCE ... 321

B.1 Overview ... 321
B.2 I/O Files ... 323
B.3 Environment Variable ... 325
B.4 Method for Manipulating ... 326
B.4.1 Command line operation ... 326
B.4.2 Subcommand file usage ... 328
B.4.3 Set options in CubeSuite+ ... 329
B.5 Option ... 330
B.5.1 Compile options ... 331
B.5.2 Assemble options ... 400
B.5.3 Link options ... 421
B.6 Specifying Multiple Options ... 476
B.6.1 Priority ... 476
B.6.2 Incompatible features ... 476
B.6.3 Dependencies ... 476
B.6.4 Relationship with #pragma directives ... 477

APPENDIX C INDEX ... 478

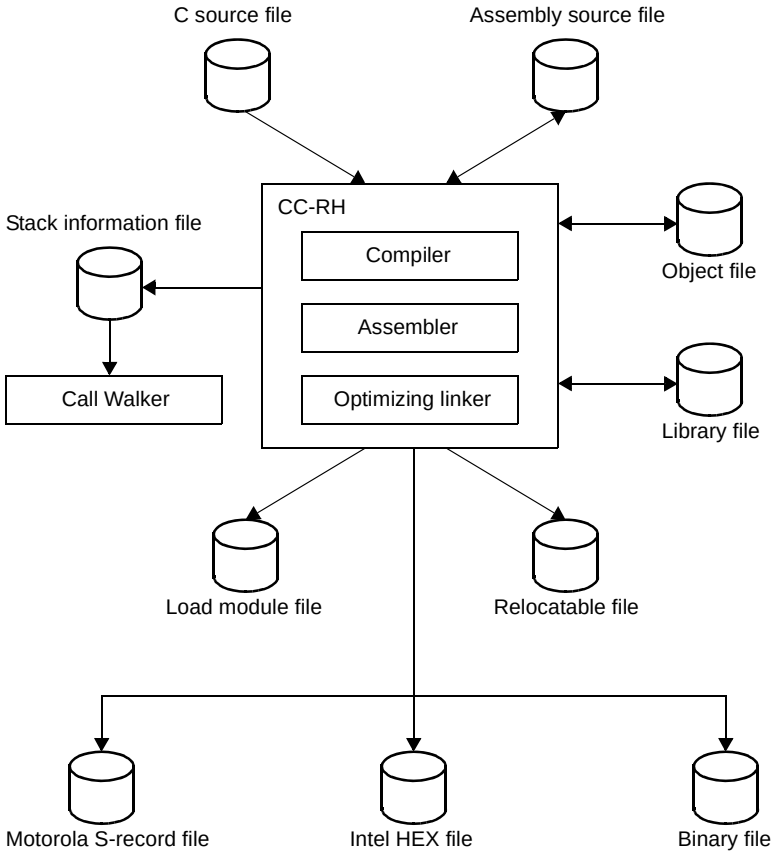
CHAPTER 1 GENERAL

This chapter explains the overview of the build tool (CC-RH).

1.1 Overview

CC-RH is comprised of components provided by this product. It enables various types of information to be configured via a GUI tool, enabling you to generate the load module file or library file, and the like from your source files, according to your objectives.

Figure 1-1. Operation Flow of CC-RH



Remark See "APPENDIX B COMMAND REFERENCE" for details about the commands that comprises CC-RH.
See "2.13 Estimate the Stack Capacity" for Call Walker.

1.2 Copyrights

This LLVM-based software was developed in compliance with the LLVM Release License. Copyrights of other software components are owned by Renesas Electronics Corporation.


```
==============================================================================
LLVM Release License
==============================================================================
University of Illinois/NCSA
Open Source License

Copyright (c) 2003-2012 University of Illinois at Urbana-Champaign.
All rights reserved.

Developed by:

    LLVM Team

    University of Illinois at Urbana-Champaign

    http://llvm.org

Permission is hereby granted, free of charge, to any person obtaining a copy of this software
and associated documentation files (the "Software"), to deal with the Software without
restriction, including without limitation the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the
Software is furnished to do so, subject to the following conditions:

    * Redistributions of source code must retain the above copyright notice, this list of
conditions and the following disclaimers.

    * Redistributions in binary form must reproduce the above copyright notice, this list of
conditions and the following disclaimers in the documentation and/or other materials provided
with the distribution.

    * Neither the names of the LLVM Team, University of Illinois at Urbana-Champaign, nor the
names of its contributors may be used to endorse or promote products derived from this
Software without specific prior written permission.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING
BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
NONINFRINGEMENT.  IN NO EVENT SHALL THE CONTRIBUTORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY
CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE,
ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS WITH THE
SOFTWARE.
```

1.3 Features

The features of CC-RH are shown below.

- High performance
  Advanced optimization is used, applying inter-module optimization as well as conventional optimization.
  This allows you to generate code that has advantages in code size and execution speed and to reduce the compilation time.

- High portability
  The C program check functions are provided to support migration from programs created for the existing compiler (SuperH RISC engine C/C++ compiler (hereafter referred to as "SHC")).

- Multifunctional
  Static analysis and other features are provided through linking to other components.

CHAPTER 2 FUNCTIONS

This chapter describes the build procedure using CubeSuite+ and about the main build functions.

2.1 Overview

This section describes how to create a load module and user library.

2.1.1 Create a load module

The procedure for creating a load module is shown below.

(1) Create or load a project

Create a new project, or load an existing one.

Remark See "CubeSuite+ Integrated Development Environment User's Manual: Start" for details about creating a new project or loading an existing one.

(2) Set a build target project

Set a build target project (see "2.11 Make Settings for Build Operations").

Remarks 1. If there is no subproject in the project, the project is always active.

2. When setting a build mode, add the build mode (see "2.11.6 Add a build mode").

(3) Set build target files

Add or remove build target files and update the dependencies (see "2.3 Set Build Target Files").

Remarks 1. See "2.7.1 Add a user library" for the method of adding a user library to the project.

2. Also, you can set the link order of object module files and library files (see "2.11.2 Set the link order of files").

(4) Specify the output of a load module

Select the type of the load module to be generated (see "2.4 Set the Type of the Output File").

(5) Set build options

Set the options for the compiler, assembler, linker, and the like (see "2.5 Set Compile Options", "2.6 Set Assemble Options", "2.7 Set Link Options").

(6) Run a build

Run a build (see "2.12 Run a Build").

The following types of builds are available.

- Build (see "2.12.1 Run a build of updated files")
- Rebuild (see "2.12.2 Run a build of all files")
- Rapid build (see "2.12.3 Run a build in parallel with other operations")
- Batch build (see "2.12.4 Run builds in batch with build modes")

Remark If there are any commands you wish to run before or after the build process, on the Property panel, from the [Common Options] tab, in the [Others] category, set the [Commands executed before build processing] and [Commands executed after build processing] properties.
If there are any commands you wish to run before or after the build process at the file level, you can set

them from the [Individual Compile Options] tab (for a C source file) and [Individual Assemble Options] tab (for an assembly source file).

(7) Save the project

Save the setting contents of the project to the project file.

Remark See "CubeSuite+ Integrated Development Environment User's Manual: Start" for details about saving the project.

2.1.2 Create a user library

The procedure for creating a user library is shown below.

(1) Create or load a project

Create a new project, or load an existing one.
When you create a new project, set a library project.

Remark See "CubeSuite+ Integrated Development Environment User's Manual: Start" for details about creating a new project or loading an existing one.

(2) Set a build target project

Set a build target project (see "2.11 Make Settings for Build Operations").

Remarks 1. If there is no subproject in the project, the project is always active.
2. When setting a build mode, add the build mode (see "2.11.6 Add a build mode").

(3) Set build target files

Add or remove build target files and update the dependencies (see "2.3 Set Build Target Files").

(4) Set build options

Set the options for the compiler, assembler, librarian, and the like (see "2.5 Set Compile Options", "2.6 Set Assemble Options", "2.9 Set Create Library Options").

Remark To create a library common to various devices, set the [Output common object file for various devices] property in the [Output File Type and Path] category from the [Common Options] tab on the Property panel.

(5) Run a build

Run a build (see "2.12 Run a Build").
The following types of builds are available.

- Build (see "2.12.1 Run a build of updated files")
- Rebuild (see "2.12.2 Run a build of all files")
- Rapid build (see "2.12.3 Run a build in parallel with other operations")
- Batch build (see "2.12.4 Run builds in batch with build modes")

Remark If there are any commands you wish to run before or after the build process, on the Property panel, from the [Common Options] tab, in the [Others] category, set the [Commands executed before build processing] and [Commands executed after build processing] properties.
If there are any commands you wish to run before or after the build process at the file level, you can set

them from the [Individual Compile Options] tab (for a C source file) and [Individual Assemble Options] tab (for an assembly source file).

(6) Save the project

Save the setting contents of the project to the project file.

Remark See "CubeSuite+ Integrated Development Environment User's Manual: Start" for details about saving the project.

2.2 Change the Build Tool Version

You can change the version of the build tool (compiler package) used in the project (main project or subproject).

Select the build tool node on the project tree and select the [Common Options] tab on the Property panel. Select [Always latest version which was installed] or the version in the [Using compiler package version] property in the [Version Select] category.

Figure 2-1. [Using compiler package version] Property

Remarks 1. When the build tool used in the main project and subprojects is the same, you can collectively change the build tool version by selecting all of the Build tool nodes and setting the property.

2. If you have selected a compiler package that has not been installed (e.g. if you open a project created in another execution environment), then that version is also displayed.

3. If the options change depending on the compiler package, then the display of the build tool's properties will change according to the selected version.

Properties that are hidden when the version is changed are saved in the project file's settings, and the values will be reproduced when the properties are displayed again.

Options are changed in accordance with the following rules. Information about changes is displayed in the Output panel.

- If you change from an older version to a newer version, the option settings will be inherited and converted (only if necessary).
- If you change from a newer version to an older version, only identical option settings will be inherited.

  Options that only exist in the older version will be set to the default values.

2.3 Set Build Target Files

Before running a build, you must add the build target files (such as C source file or assembly source file) to the project. This section explains operations on setting files in the project.

2.3.1 Add a file to a project

Files can be added to a project by the following methods.

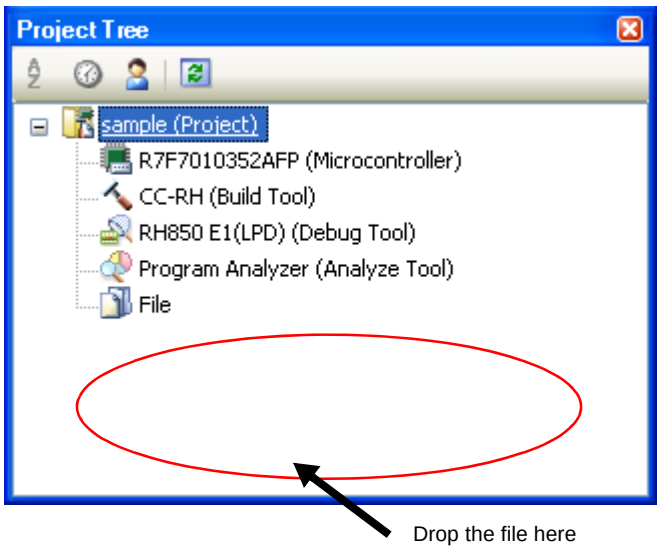
- Adding an existing file
- Creating and adding an empty file

(1) Adding an existing file

(a) Add an individual files

Drag the file from Explorer or the like, and drop it onto the empty space below the project tree. The file is added below the File node.

Figure 2-2. Project Tree Panel (File Drop Location)



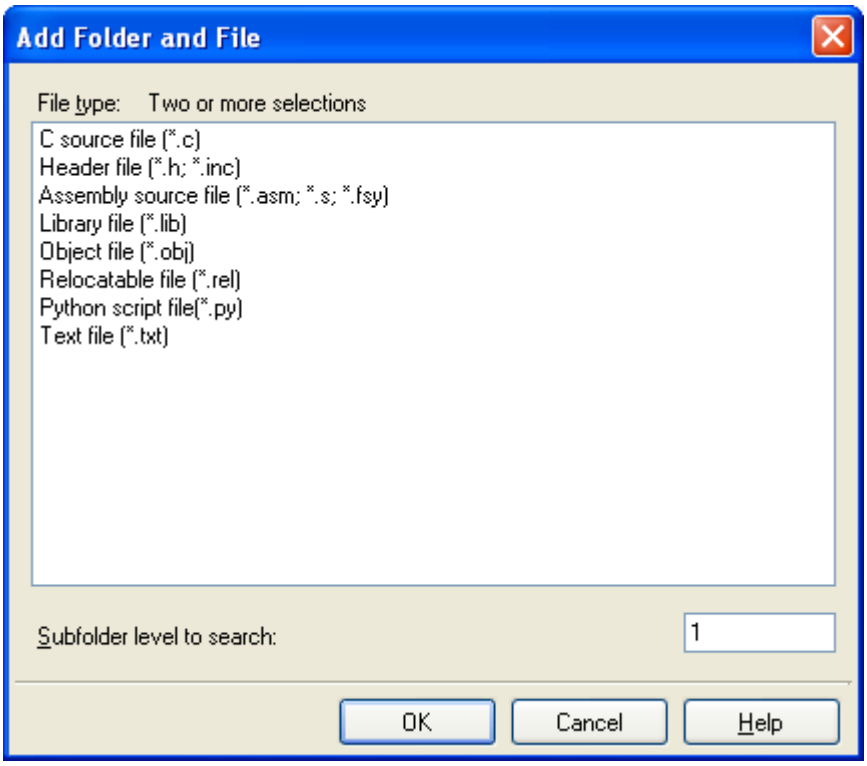
(b) Add a folder

Drag the folder from Explorer or the like, and drop it onto the empty space below the project tree. The Add Folder and File dialog box will open.

Remark You can also add multiple folders to the project at the same time by dragging multiple folders at same time and dropping them onto the project tree.

Caution **When the folder with the name that is more than 200 characters is dropped, the folder is added to the project tree as a category with the name that 201st character and after are deleted.**

Figure 2-3. Add Folder and File Dialog Box



In the dialog box, select the types of the files to be added to the project and specify the number of levels of the subfolder to be added to the project. And then click the [OK] button.

Remark You can select multiple file types by left clicking while holding down the [Ctrl] or [Shift] key.
If nothing is selected, it is assumed that all types are selected.

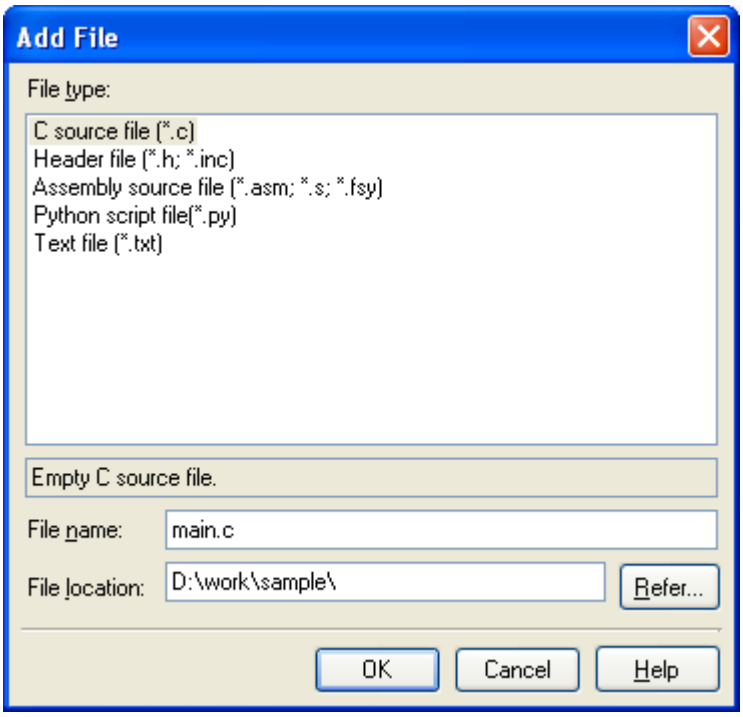
The folder is added below the File node.
Note that on the project tree, the folder is the category.

Remark When the category node created by the user exists, you can add a file below the node by dropping the file onto the node (see "2.3.4 Classify a file into a category" for a category node).

(2) Creating and adding an empty file

On the project tree, select either one of the Project node, Subproject node, or File node, and then select [Add] >> [Add New File...] from the context menu. The Add File dialog box will open.

Figure 2-4. Add File Dialog Box



In the dialog box, specify the file to be created and then click the [OK] button.
The file is added below the File node.

The project tree after adding the file and folder will look like the one below.

Figure 2-5. Project Tree Panel (After Adding File "main.c")

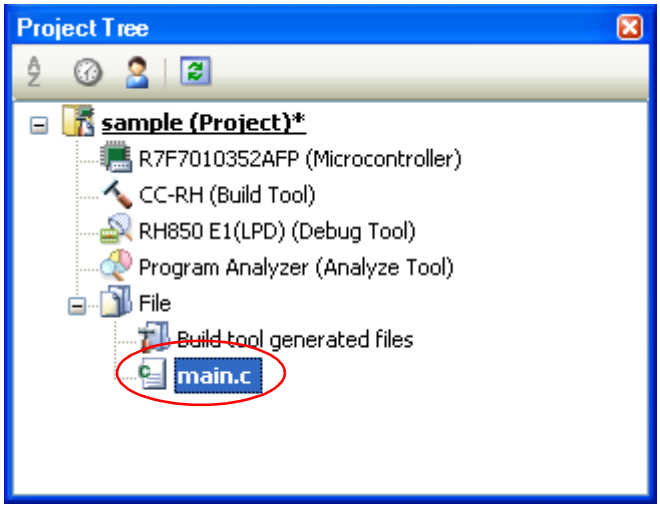
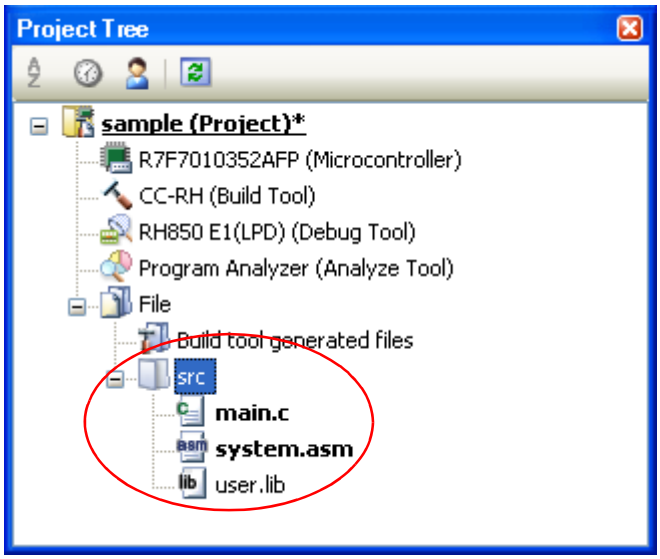


Figure 2-6. Project Tree Panel (After Adding Folder "src")



Remark The location of the file added below the File node depends on the current file display order setting.
See "2.3.5 Change the file display order" for the method of changing the file display order.

Cautions 1. If the paths differ, you can add source files with the same name.
Note, however, that if the setting of the output file name is left as the default, the output files will have the same name, which will prevent the build from running correctly (for example, when adding D:\sample1\func.c and D:\sample2\func.c, the default output file name for these files is both func.obj).
To avoid this problems, set the output file name for each of those files to a different name with the individual build options for the source files.
Changing the name of the C source file is made with the [Object module file name] property in the [Output File] category from the [Individual Compile Options] tab.
The changing the name of the assembly source file is made with the [Object file name] property in the [Output File] category from the [Individual Assemble Options] tab.
See "2.10.2 Set build options at the file level" for how to set the individual build options.

2. If source files with the same name are added, the target file cannot opened during debugging.

3. Up to 5000 files can be added to the main project or subproject.

When a new file is added, an empty file is created in the location specified in the Add File dialog box.
By double clicking the file name on the project tree, you can open the Editor panel and edit the file.
The files that can be opened with the Editor panel are shown below.

- C source file (*.c)
- Assembly source file (*.asm, *.s)
- Symbol address file (*.fsy)
- Header file (*.h, *.inc)
- List file for assembly program (*.lst, *.lis)
- Preprocessor expanded file for assembly program (*.exp)
- Assemble list file (*.prn)
- Link order specification file (*.mtls)
- Link map file (*.map)
- Library list file (*.lbp)
- Intel Hex file (*.hex)

- Motorola S-record file (*.mot)
- Text file (*.txt)

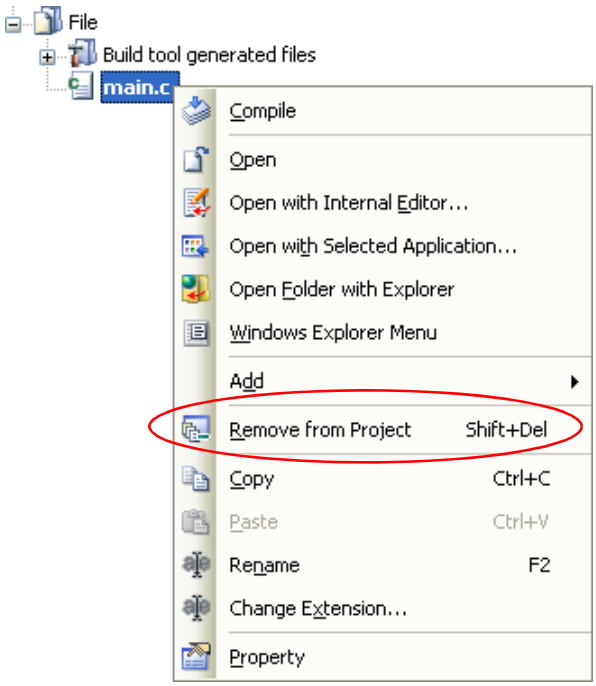
Remarks 1. You can use one of the procedures below to open files other than those listed above in the Editor panel.
- Drag a file and drop it onto the Editor panel.
- Select a file and then select [Open with Internal Editor...] from the context menu.

2. When the environment is set to use an external text editor on the Option dialog box, the file is opened with the external text editor that has been set.
Other files are opened with the applications associated by the host OS.

2.3.2 Remove a file from a project

To remove the file added to a project, select the file to be removed from the project on the project tree and then select [Remove from Project] from the context menu.

Figure 2-7. [Remove from Project] Item



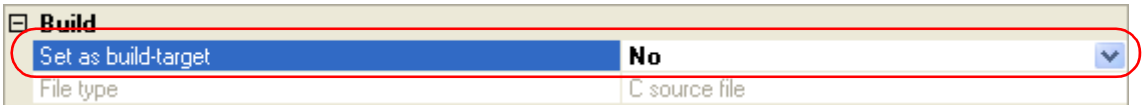
2.3.3 Remove a file from the build target

You can remove the specific file from the build target out of all the files added to the project.

Select the file to be removed from the build target on the project tree and select the [Build Settings] tab on the Property panel.

Select [No] on the [Set as build-target] property in the [Build] category.

Figure 2-8. [Set as build-target] Property



Remark The files that this function can be applied to are C source files, assembly source files, object files, and library files.

2.3.4 Classify a file into a category

You can create a category under the File node and classify files by the category. This makes it easier to view files added to the project on the project tree, and makes it easier to manage files according to function.

To create a category node, select either one of the Project node, Subproject node, or File node on the project tree, and then select [Add] >> [Add New File...] from the context menu.

Figure 2-9. [Add New Category] Item (For File Node)

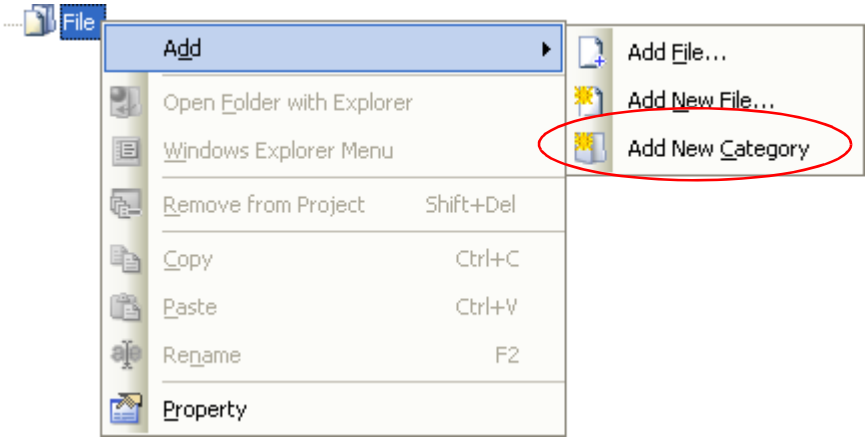
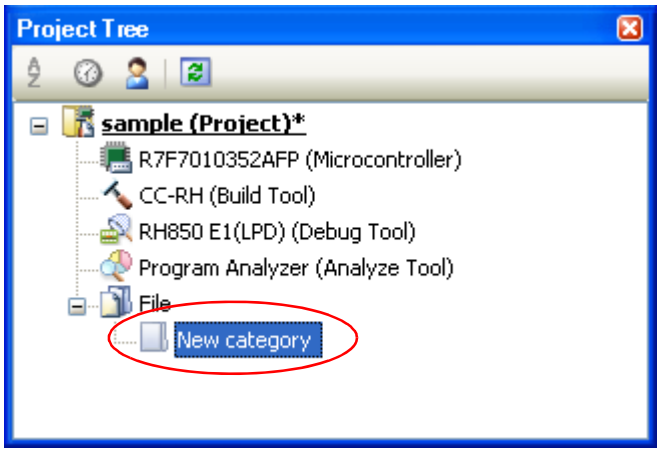


Figure 2-10. Project Tree Panel (After Adding Category Node)



Remarks 1. The default category name is "New category".
To change the category name, you can use [Rename] from the context menu of the category node.

2. You can also add a category node with the same name as the existing category node.

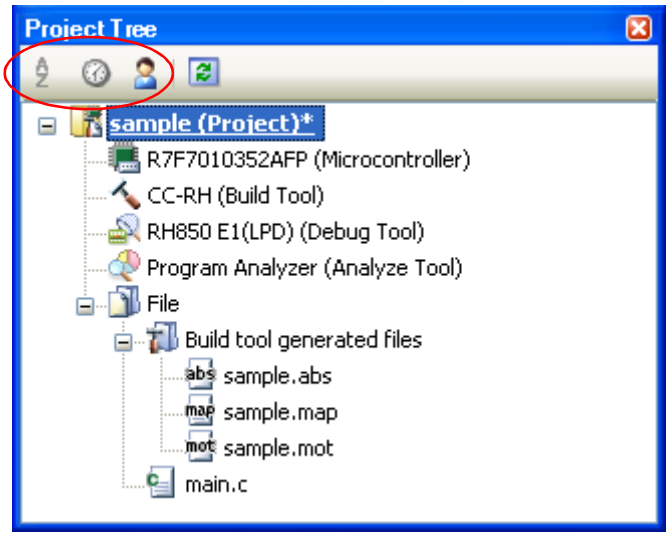
3. Categories can be nested up to 20 levels.

You can classify files into the created category node by dragging and dropping the file.

2.3.5 Change the file display order

You can change the display order of the files and category nodes by the buttons on the project tree.

Figure 2-11. Toolbar (Project Tree Panel)



Select any of the buttons below on the toolbar of the Project Tree panel.

Button	Description
	Sorts category nodes and files in order of their names. : Ascending order : Descending order : Ascending order
	Sorts category nodes and files in order of their timestamp. : Descending order : Ascending order : Descending order
	Displays category nodes and files in order of the user definition (default). You can change the display order of category nodes and files arbitrarily by dragging and dropping them.

2.3.6 Update file dependencies

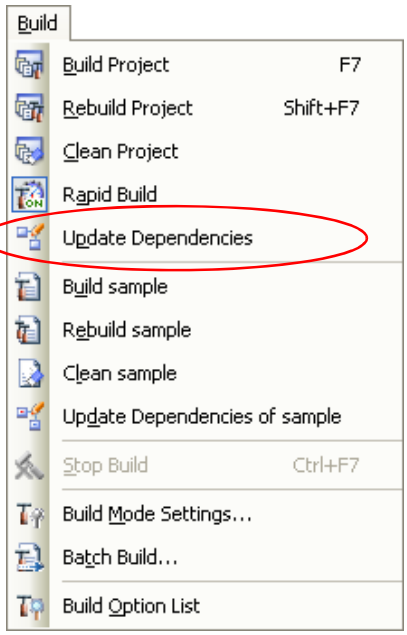
When you perform a change (changing include file paths, adding the include statement of the header file to the source file, etc.) that effects the file dependencies in the compile option settings or assemble option settings, you must update the dependencies of the relevant files.

Updating file dependencies is performed for the entire project (main project and subprojects) or active project.

(1) For the entire project

From the [Build] menu, select [Update Dependencies].

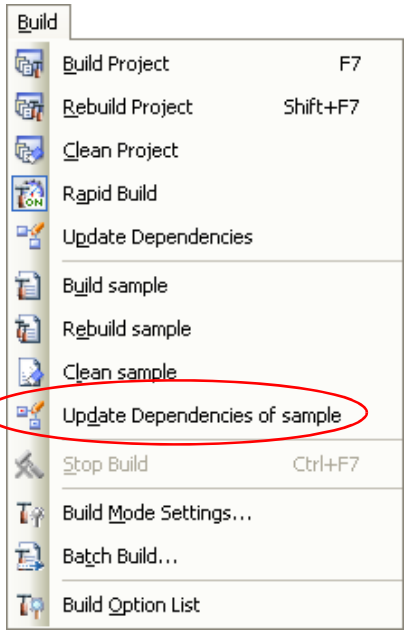
Figure 2-12. [Update Dependencies] Item



(2) For the active project

From the [Build] menu, select [Update Dependencies of *active project*].

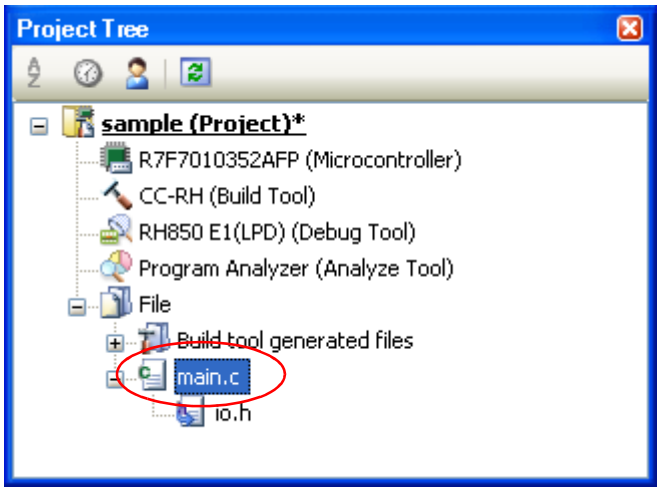
Figure 2-13. [Update Dependencies of *active project*] Item



Remark If there are files being edited with the Editor panel when updating file dependencies, then all these files are saved.

Dependency files (include files) can be displayed under the source file on the project tree.

Figure 2-14. Project Tree Panel (After Displaying Dependency File)



The display of the dependency files is updated on the following timings:

- When the first build is run after the project is loaded
- When on the toolbar is clicked
- When [Update Dependencies] is selected from the [Build] menu
- When [Update Dependencies of *active project*] is selected from the [Build] menu

Remarks 1. The display of dependency files is valid only when the [Show dependency files in project tree] checkbox in the [General - Build/Debug] category of the Option dialog box is selected.

2. Information on the dependency files displayed on the project tree is not saved in the project file.

Cautions 1. **When checking for dependences on include files, CubeSuite+ does not support cases of include files to which conditional statements such as #if apply or for which the #include directive is commented out.**
Therefore, there is a case where this product regards an include file unnecessary for a build as a necessary file (In the example below, header1.h and header5.h are judged as required for build).

```
#if         0
#include    "header1.h"     /* Dependence relationship judged to exist */
#else                       /* ! zero */
#include    "header2.h"     /* Dependence relationship to exist */
#endif

#define     AAA
#ifdef      AAA
#include    "header3.h"     /* Dependence relationship to exist */
#else
#include    "header4.h"     /* Dependence relationship to exist */
#endif

/*
#include    "header5.h"     /* Dependence relationship judged to exist */
*/
```

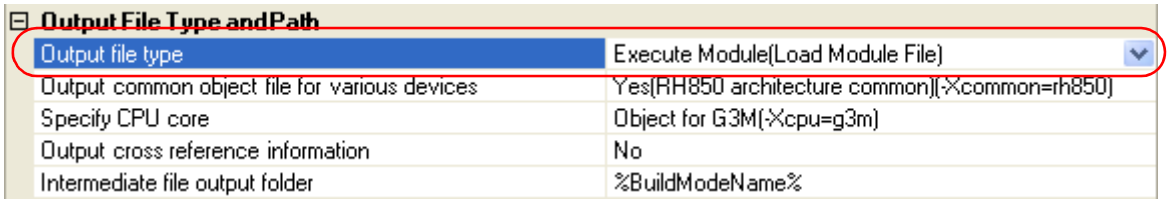
2. **When checking for dependences on include files, CubeSuite+ does not support include statements that follow comments or comment marks that are on the same line.**
Therefore, there is a case where this product regards an include file necessary for a build as a unnecessary file (In the example below, header6.h and header7.h are judged as no-required for build).

```
/* comment */   #include    "header6.h"     /* Dependence relationship judged not
to exist */


/*

comment

*/  #include    "header7.h"                 /* Dependence relationship judged not
to exist */
```

2.4 Set the Type of the Output File

Set the type of the file to be output as the product of the build.
Select the build tool node on the project tree and select the [Common Options] tab on the Property panel.
Select the file type in the [Output file type] property in the [Output File Type and Path] category.

Figure 2-15. [Output file type] Property



(1) When [Execute Module(Load Module File)] is selected (Default)

A load module file is generated.
The load module file will be the debug target.

(2) When [Execute Module(Hex File)] is selected

A hex file is generated.
The hex file will be the debug target.

Caution For the library project, this property is always [Library] and cannot be changed.

2.4.1 Change the output file name

The names of the load module file, hex file, and library file output by the build tool are set as follows by default.

Load module file name: %ProjectName%.abs
Hex file name: %ProjectName%.mot
Library file name: lib%ProjectName%.lib

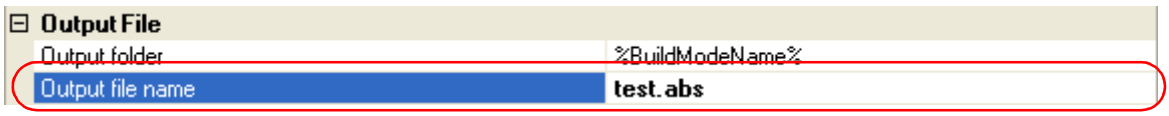
Remark "%ProjectName%" is a placeholder. It is replaced with the project name.

The method to change these file names is shown below.

(1) When changing the load module file name and non-ROMized load module file name

Select the build tool node on the project tree and select the [Link Options] tab on the Property panel.
Enter the file name to be changed to in the [Output file name] property in the [Output File] category.

Figure 2-16. [Output file name] Property



This property supports the following placeholders.

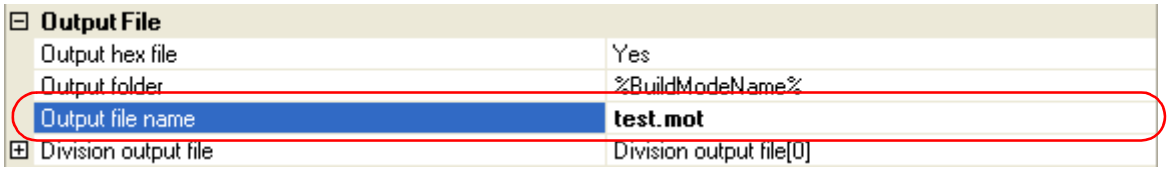
%ActiveProjectName%: Replaces with the active project name.
%MainProjectName%: Replaces with the main project name.
%ProjectName%: Replaces with the project name.

Remark You can also change the option in the same way with the [Output file name] property in the [Frequently Used Options(for Link)] category on the [Common Options] tab.

(2) When changing the hex file name

Select the build tool node on the project tree and select the [Hex Output Options] tab on the Property panel. Enter the hex file name to be changed to in the [Output file name] property in the [Output File] category.

Figure 2-17. [Output file name] Property



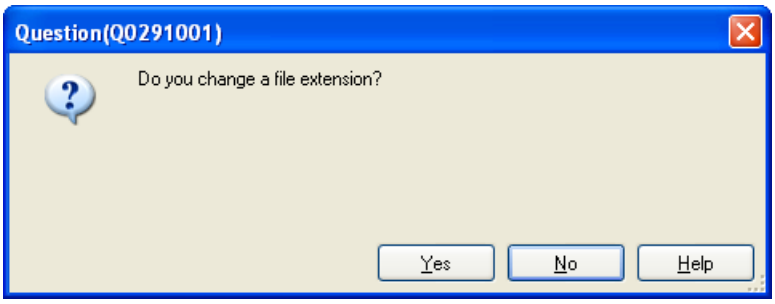
This property supports the following placeholders.

%ActiveProjectName%: Replaces with the active project name.
%MainProjectName%: Replaces with the main project name.
%ProjectName%: Replaces with the project name.

Remark You can also change the option in the same way with the [Output file name] property in the [Frequently Used Options(for Hex Output)] category on the [Common Options] tab.

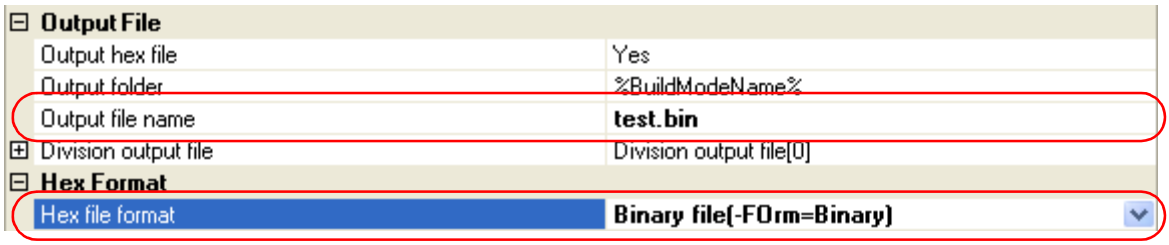
If the [Hex file format] property in the [Hex Format] category is changed, the following message dialog box will open.

Figure 2-18. Message Dialog Box



When [Yes] is selected in the dialog box, the extension of the output file name is changed according to the format selected in the [Hex file format] property.

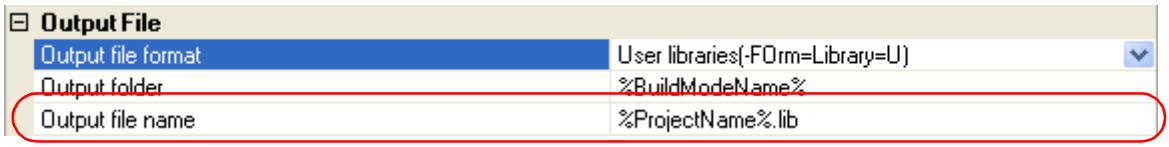
Figure 2-19. [Output file name] and [Hex file format] Property



(3) When changing the library file name

Select the build tool node on the project tree and select the [Create Library Options] tab on the Property panel. Enter the library file name to be changed to on the [Output file name] property in the [Output File] category.

Figure 2-20. [Output file name] Property



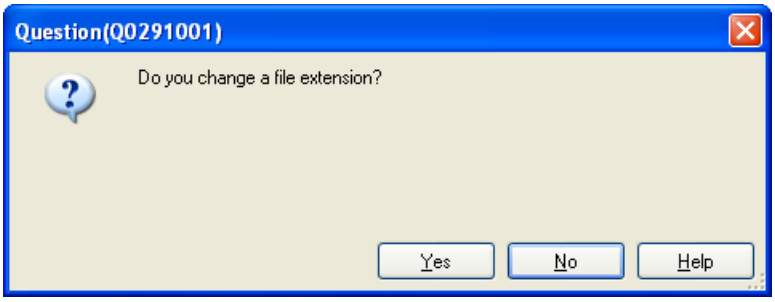
This property supports the following placeholders.

%ActiveProjectName%: Replaces with the active project name.
%MainProjectName%: Replaces with the main project name.
%ProjectName%: Replaces with the project name.

Remark You can also change the option in the same way with the [Output file name] property in the [Frequently Used Options(for Create Library)] category on the [Common Options] tab.

If the [Output file format] property is changed, the following message dialog box will open.

Figure 2-21. Message Dialog Box



When [Yes] is selected in the dialog box, the extension of the output file name is changed according to the format selected in the [Output file format] property.

Figure 2-22. [Output file format] and [Output file name] Property

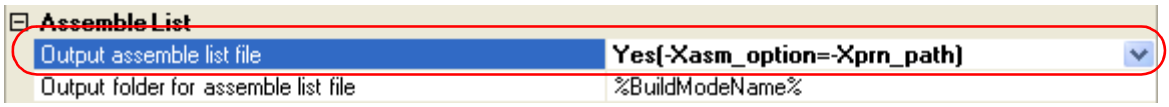
2.4.2 Output an assemble list

The assemble list (the code of the assemble result) is output to the assemble list file.

Select the build tool node on the project tree and select the [Compile Options] tab on the Property panel.

To output the assemble list file, select [Yes(-Xasm_option=-Xprn_path)] in the [Output assemble list file] property in the [Assemble List] category.

Figure 2-23. [Output assemble list file] Property



When outputting the assemble list file, you can set the output folder and output file name.

(1) Set the output folder

Setting the output folder is made with the [Output folder for assemble list file] property by directly entering in the text box or by the [...] button.

This property supports the following placeholder.

%BuildModeName%: Replaces with the build mode name.

"%BuildModeName%" is set by default.

The file name will be the source file name with the extension replaced by ".prn".

Remark See "3.1 Assemble List File" for details about the assemble list file.

2.4.3 Output map information

The map information (the information of the link result) is output to the link map file.

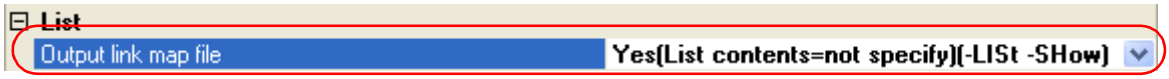
Select the build tool node on the project tree and select the [Link Options] tab on the Property panel.

To output the link map file, set the [Output link map file] property in the [List] category.

(1) Output information according to the output format

Select [Yes(List contents=not specify)(-LISt -SHow)] or [Yes(List contents=ALL)(-LISt -SHow=ALL)] in the [Output link map file] property.

Figure 2-24. [Output link map file] Property (When Information According To Output Format Is Output)



Remark See [Remark] in "-SHow" for differences between the -SHow and -SHow=ALL options.

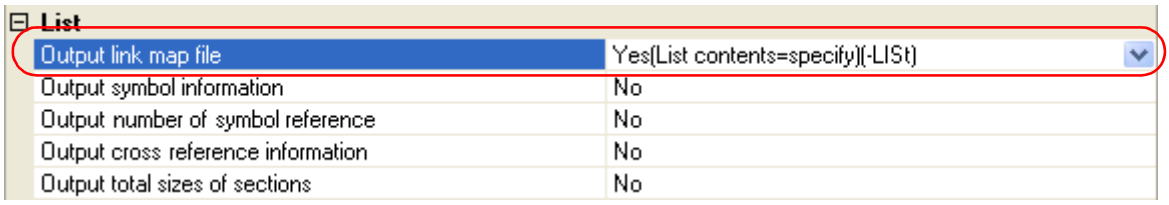
(2) Specify information to be output

Select [Yes(List contents=specify)(-LISt)] in the [Output link map file] property. The following property will be displayed.

- [Output symbol information] property
- [Output number of symbol reference] property
- [Output cross reference information] property
- [Output total sizes of sections] property

Select [Yes] for each output information property.

Figure 2-25. [Output link map file] Property (When Information To Be Output Is Specified)



The link map file is output to the project folder.
It is also shown on the project tree, under the Build tool generated files node.
~~The file name will be the output file name with the extension replaced by ".map".~~
The file name will be the project file name with the extension replaced by ".map".

Remark See "3.2 Link Map File" for details about the link map file.

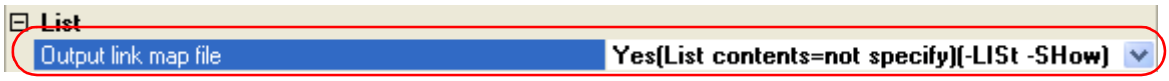
2.4.4 Output library information

The library information (information from the library creation result) is output to the library list file.
Select the build tool node on the project tree and select the [Create Library Options] tab on the Property panel.
To output the library list file, set the [Output link map file] property in the [List] category.

(1) Output information according to the output format

Select [Yes(List contents=not specify)(-LISt -SHow)] or [Yes(List contents=ALL)(-LISt -SHow=ALL)] in the [Output link map file] property.

Figure 2-26. [Output link map file] Property (When Information According To Output Format Is Output)



Remark See [Remark] in "-SHow" for differences between the -SHow and -SHow=ALL options.

(2) Specify information to be output

Select [Yes(List contents=specify)(-LISt)] in the [Output link map file] property. The following property will be displayed.

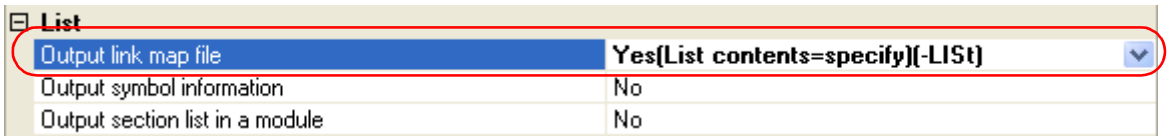
- [Output symbol information] property
- [Output section list in a module] property**Note 1**
- [Output cross reference information] property**Note 2**
- [Output total sizes of sections] property**Note 2**

Notes 1. This property is displayed only when [User libraries(-FOrm=Library=U)] or [System libraries(-FOrm=Library=S)] in the [Output file format] property in the [Output File] category is selected.

2. This property is displayed only when [Relocate file(-FOrm=Relocate)] in the [Output file format] property in the [Output File] category is selected.

Select [Yes] for each output information property.

Figure 2-27. [Output link map file] Property (When Information To Be Output Is Specified)



The library list file is output to the project folder.

It is also shown on the project tree, under the Build tool generated files node.

~~The file name will be the output file name with the extension replaced by ".lbp".~~

The file name will be the project file name with the extension replaced by ".lbp".

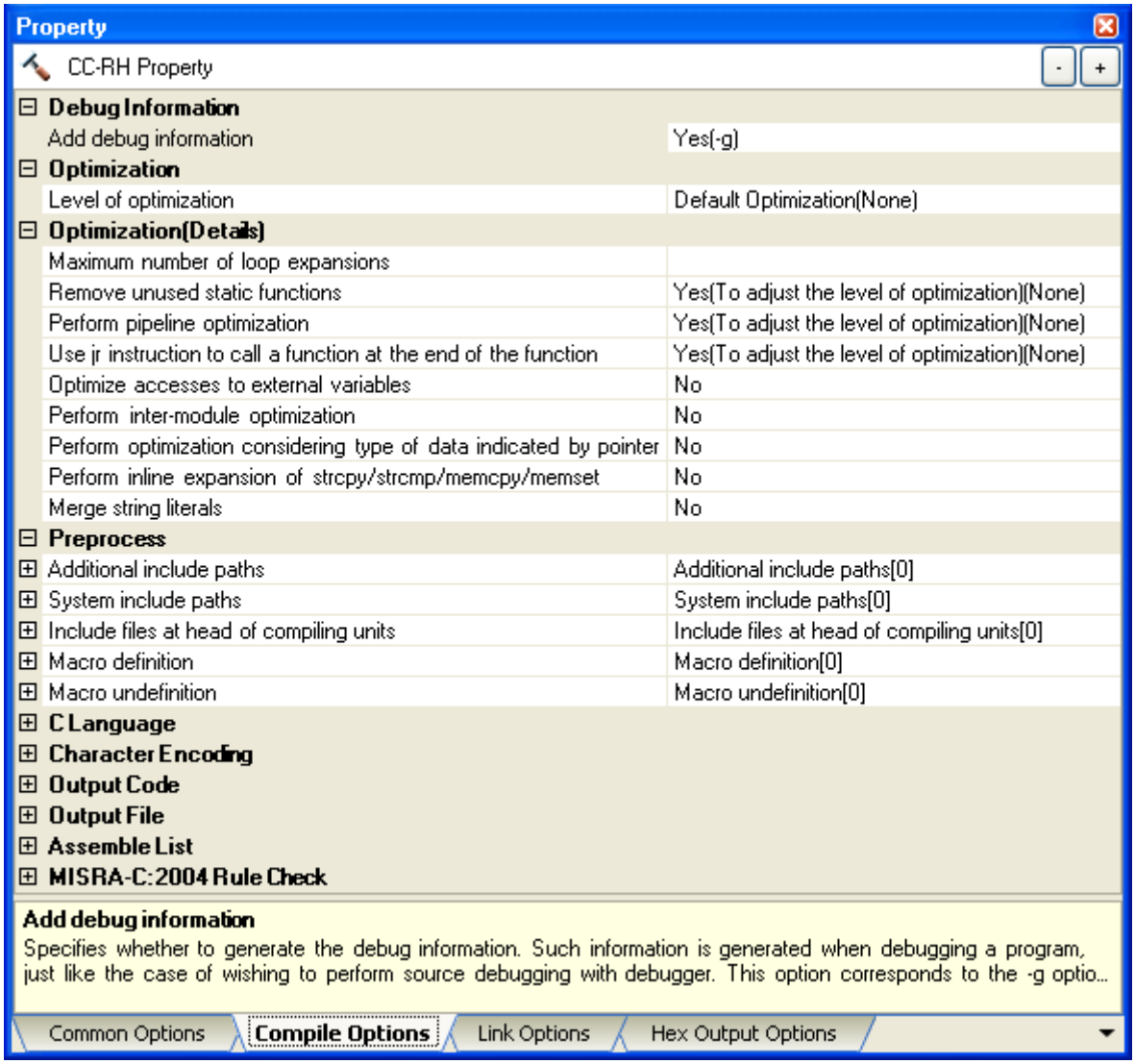
Remark See "3.4 Library List File" for details about the library list file.

2.5 Set Compile Options

To set options for the compile phase, select the Build tool node on the project tree and select the [Compile Options] tab on the Property panel.

You can set the various compile options by setting the necessary properties in this tab.

Figure 2-28. Property Panel: [Compile Options] Tab



Remark Often used options have been gathered under the [Frequently Used Options(for Compile)] category on the [Common Options] tab.

2.5.1 Perform optimization with the code size precedence

Select the build tool node on the project tree and select the [Compile Options] tab on the Property panel.

To perform optimization with the code size precedence, select [Code Size Precedence(-Osize)] in the [Optimization Level] property in the [Optimization] category.

Figure 2-29. [Level of optimization] Property (Code Size Precedence)

Remark You can also set the option in the same way with the [Optimization Level] property in the [Frequently Used Options(for Compile)] category on the [Common Options] tab.

2.5.2 Perform optimization with the execution speed precedence

Select the build tool node on the project tree and select the [Compile Options] tab on the Property panel.

To perform optimization with the execution speed precedence, select [Speed Precedence(-Ospeed)] in the [Optimization Level] property in the [Optimization] category.

Figure 2-30. [Level of optimization] Property (Execution Speed Precedence)

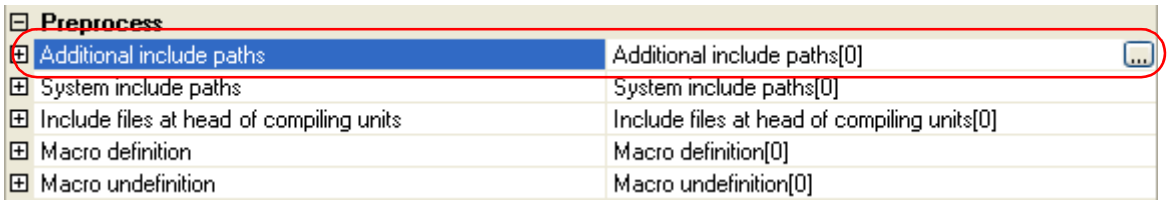
Remark You can also set the option in the same way with the [Optimization Level] property in the [Frequently Used Options(for Compile)] category on the [Common Options] tab.

2.5.3 Add an include path

Select the build tool node on the project tree and select the [Compile Options] tab on the Property panel.

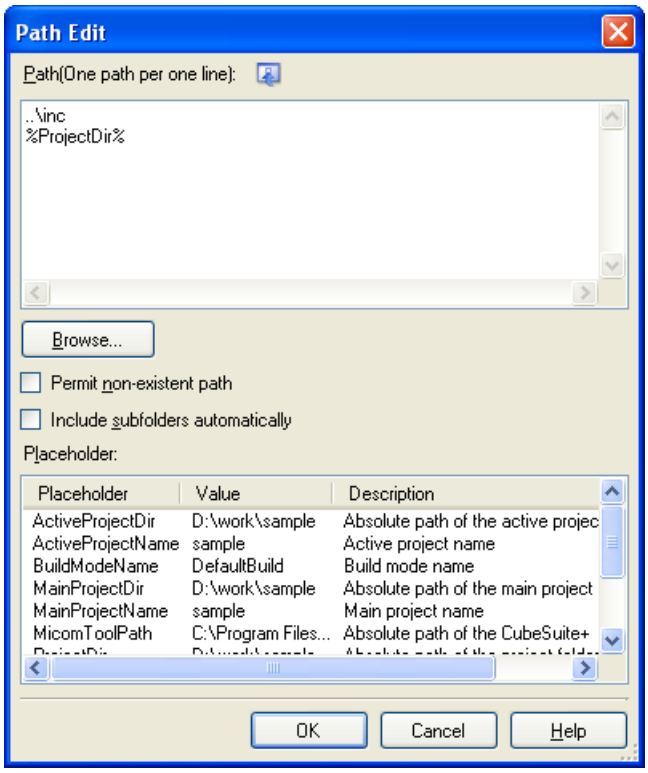
The include path setting is made with the [Additional include paths] property in the [Preprocess] category.

Figure 2-31. [Additional include paths] Property



If you click the [...] button, the Path Edit dialog box will open.

Figure 2-32. Path Edit Dialog Box



Enter the include path per line in [Path(One path per one line)].
You can specify up to 259 characters per line, up to 256 lines.

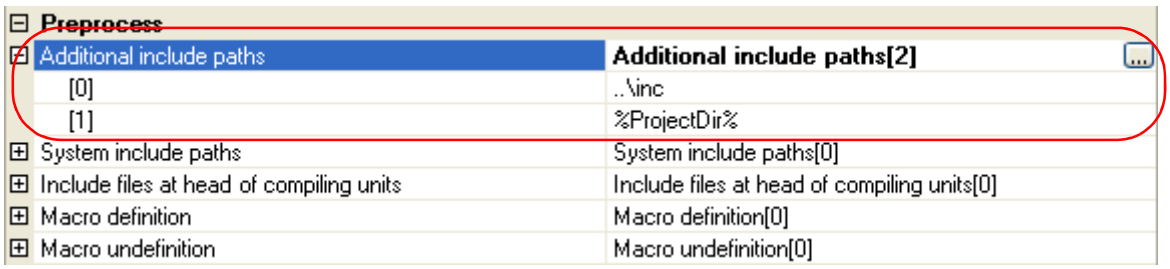
Remarks 1. This property supports placeholders.
If a line is double clicked in [Placeholder], the placeholder will be reflected in [Path(One path per one line)].

2. You can also specify the include path by one of the following procedures.
- Drag and drop the folder using such as Explorer.
- Click the [Browse...] button, and then select the folder in the Browse For Folder dialog box.
- Double click a row in [Placeholder].

3. Select the [Include subfolders automatically] check box before clicking the [Browse...] button to add all paths under the specified one (down to 5 levels) to [Path(One path per one line)].

If you click the [OK] button, the entered include paths are displayed as subproperties.

Figure 2-33. [Additional include paths] Property (After Adding Include Paths)



To change the include paths, you can use the [...] button or enter the path directly in the text box of the subproperty.

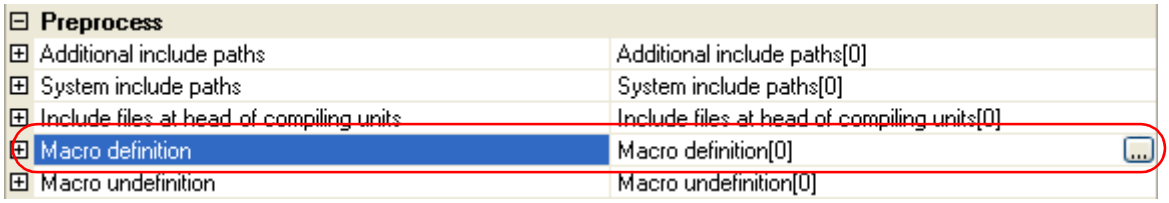
When the include path is added to the project tree, the path is added to the top of the subproperties automatically.

Remark You can also set the option in the same way with the [Additional include paths] property in the [Frequently Used Options(for Compile)] category on the [Common Options] tab.

2.5.4 Set a macro definition

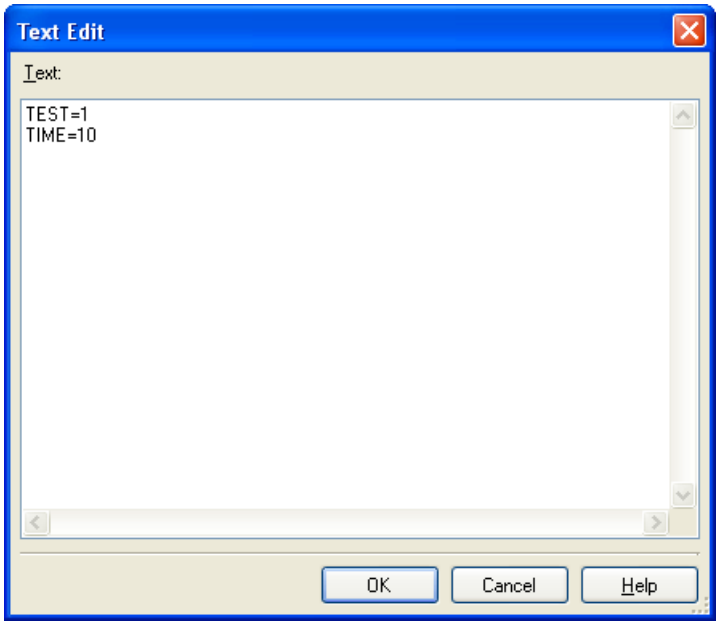
Select the build tool node on the project tree and select the [Compile Options] tab on the Property panel.
The macro definition setting is made with the [Macro definition] property in the [Preprocess] category.

Figure 2-34. [Macro definition] Property



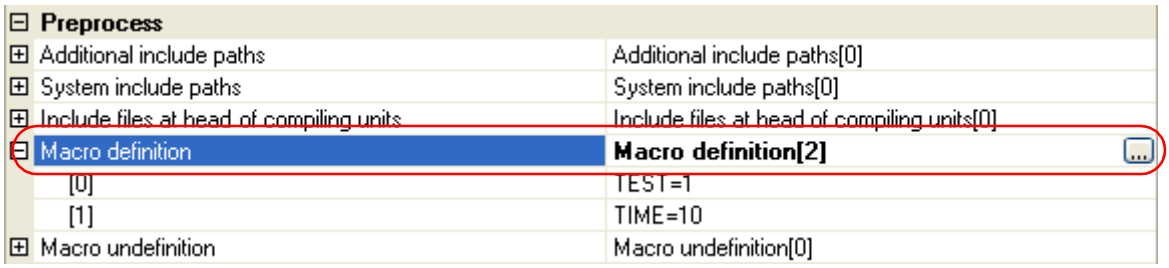
If you click the [...] button, the Text Edit dialog box will open.

Figure 2-35. Text Edit Dialog Box



Enter the macro definition in [Text] in the format of "*macro name=defined value*", with one macro name per line.
You can specify up to 256 characters per line, up to 256 lines.
The "*=defined value*" part can be omitted, and in this case, "1" is used as the defined value.
If you click the [OK] button, the entered macro definitions are displayed as subproperties.

Figure 2-36. [Macro definition] Property (After Setting Macros)



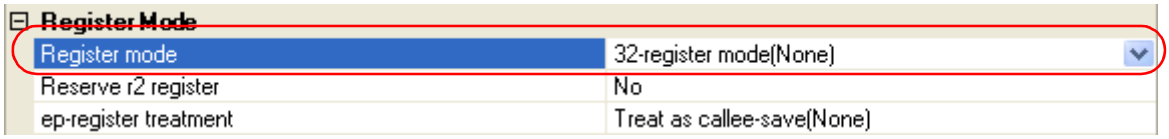
To change the macro definitions, you can use the [...] button or enter the path directly in the text box of the subproperty.

Remark You can also set the option in the same way with the [Macro definition] property in the [Frequently Used Options(for Compile)] category on the [Common Options] tab.

2.5.5 Change the register mode

Select the build tool node on the project tree and select the [Common Options] tab on the Property panel.
Select the register mode to on the [Register mode] property in the [Register Mode] category.

Figure 2-37. [Register mode] Property



You can select from the following register modes.

Register Mode	Working Registers	Registers for Register Variables
32-register mode(None)	r10 to r19	r20 to r29
22-register mode(-Xreg_mode=22)	r10 to r14	r25 to r29
Universal register mode(-Xreg_mode=common)	r10 to r14	r25 to r29

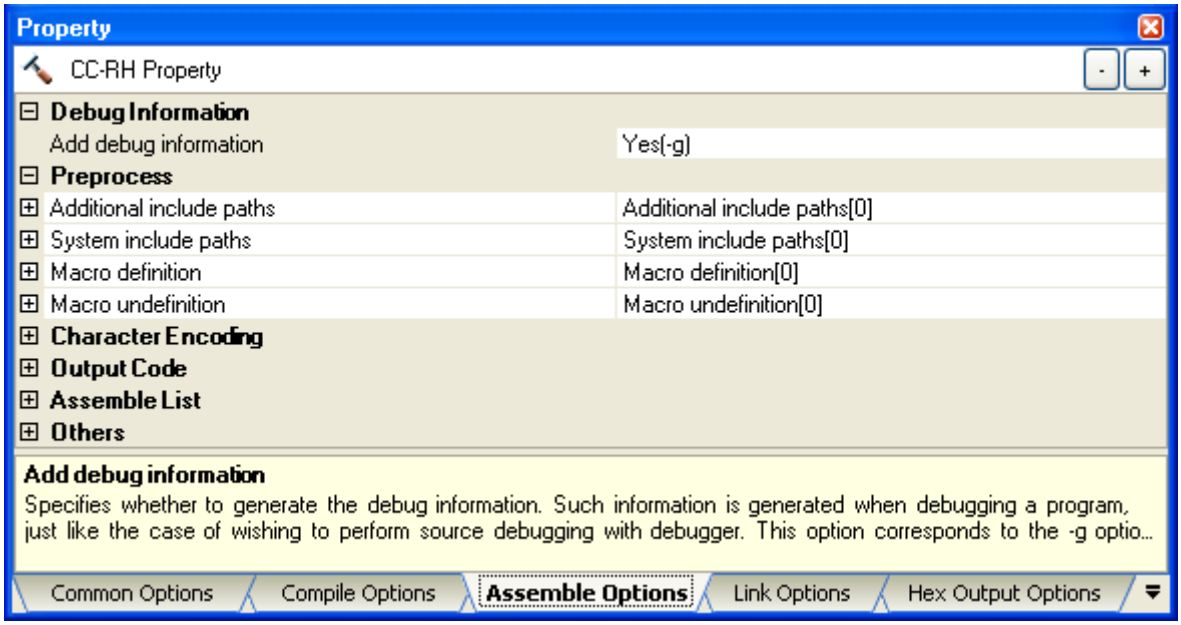
Remark See "CubeSuite+ Integrated Development Environment User's Manual: RH850 Coding" for details about register modes.

2.6 Set Assemble Options

To set options for the assemble phase, select the Build tool node on the project tree and select the [Assemble Options] tab on the Property panel.

You can set the various assemble options by setting the necessary properties in this tab.

Figure 2-38. Property Panel: [Assemble Options] Tab



Remark Often used options have been gathered under the [Frequently Used Options(for Assemble)] category on the [Common Options] tab.

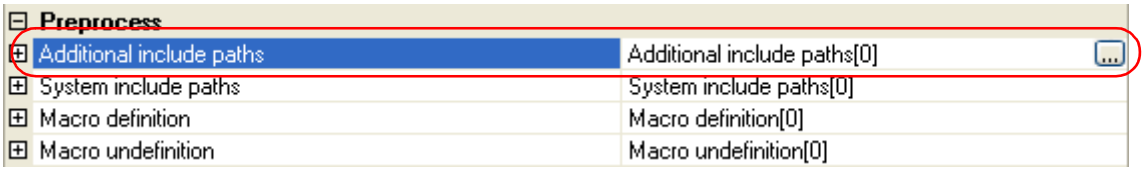
Caution **This tab is displayed only when [No] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected.**

2.6.1 Add an include path

Select the build tool node on the project tree and select the [Assemble Options] tab on the Property panel.

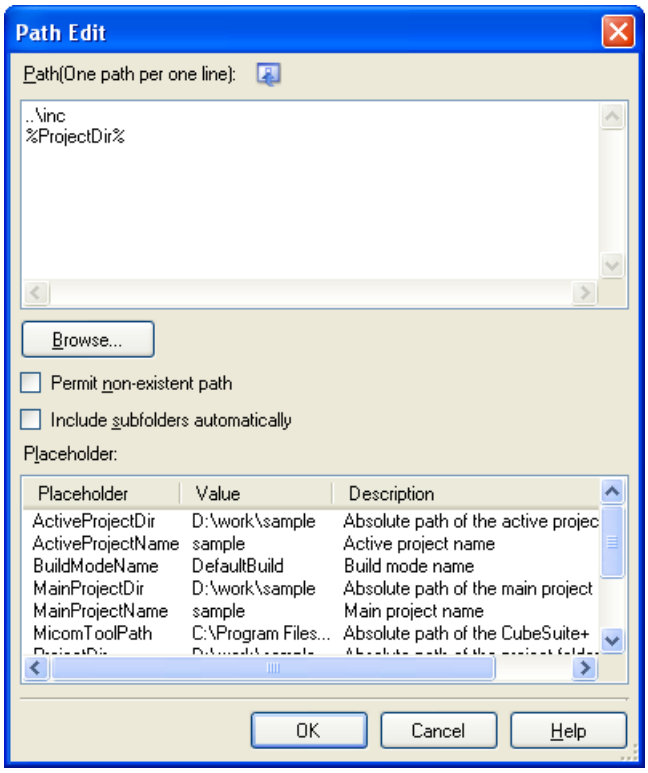
The include path setting is made with the [Additional include paths] property in the [Preprocess] category.

Figure 2-39. [Additional include paths] Property



If you click the [...] button, the Path Edit dialog box will open.

Figure 2-40. Path Edit Dialog Box



Enter the include path per line in [Path(One path per one line)].
You can specify up to 259 characters per line, up to 256 lines.

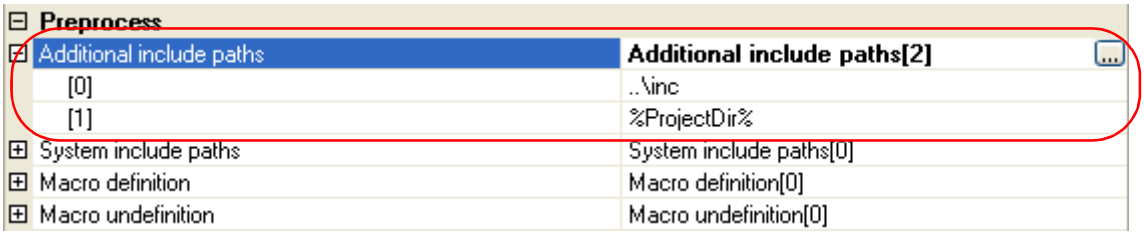
Remarks 1. This property supports placeholders.
If a line is double clicked in [Placeholder], the placeholder will be reflected in [Path(One path per one line)].

2. You can also specify the include path by one of the following procedures.
- Drag and drop the folder using such as Explorer.
- Click the [Browse...] button, and then select the folder in the Browse For Folder dialog box.
- Double click a row in [Placeholder].

3. Select the [Include subfolders automatically] check box before clicking the [Browse...] button to add all paths under the specified one (down to 5 levels) to [Path(One path per one line)].

If you click the [OK] button, the entered include paths are displayed as subproperties.

Figure 2-41. [Additional include paths] Property (After Adding Include Paths)



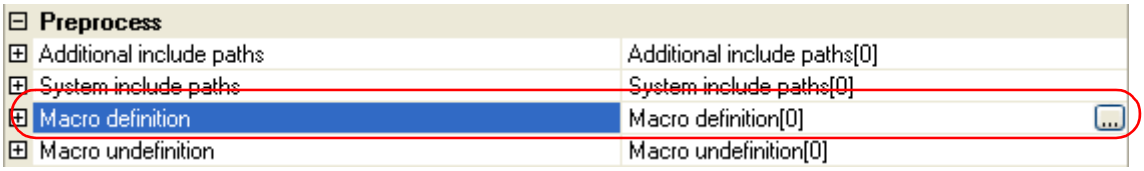
To change the include paths, you can use the [...] button or enter the path directly in the text box of the subproperty.
When the include path is added to the project tree, the path is added to the top of the subproperties automatically.

Remark You can also set the option in the same way with the [Additional include paths] property in the [Frequently Used Options(for Assemble)] category on the [Common Options] tab.

2.6.2 Set a macro definition

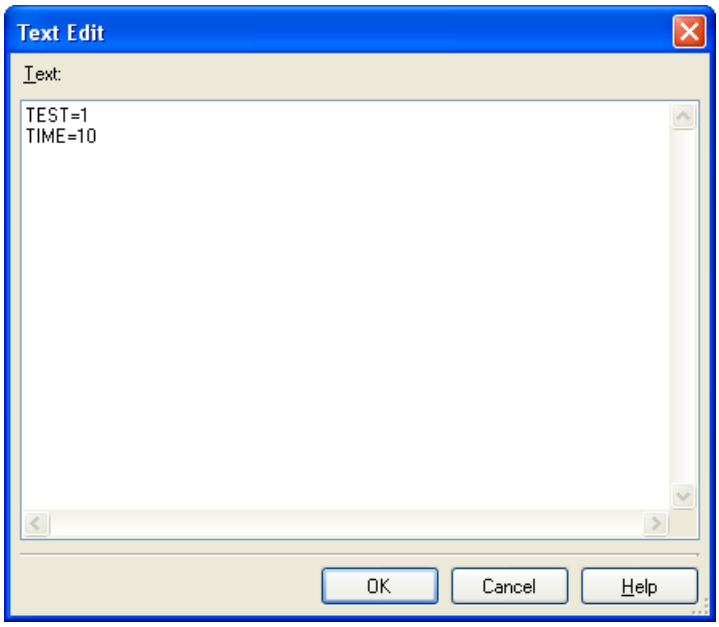
Select the build tool node on the project tree and select the [Assemble Options] tab on the Property panel.
The macro definition setting is made with the [Macro definition] property in the [Preprocess] category.

Figure 2-42. [Macro definition] Property



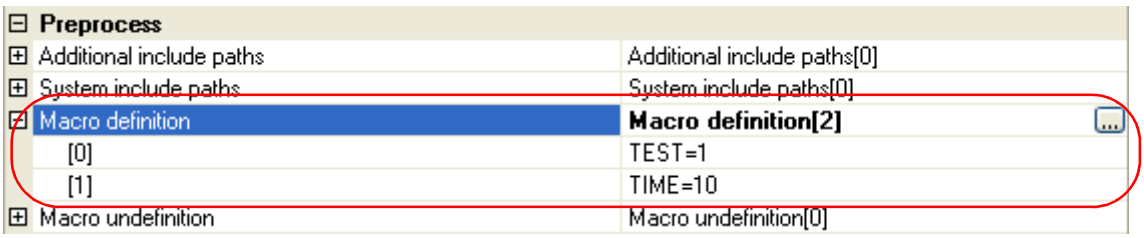
If you click the [...] button, the Text Edit dialog box will open.

Figure 2-43. Text Edit Dialog Box



Enter the macro definition in [Text] in the format of "*macro name=defined value*", with one macro name per line.
You can specify up to 256 characters per line, up to 256 lines.
The "=*defined value*" part can be omitted, and in this case, "1" is used as the defined value.
If you click the [OK] button, the entered macro definitions are displayed as subproperties.

Figure 2-44. [Macro definition] Property (After Setting Macros)



To change the macro definitions, you can use the [...] button or enter the path directly in the text box of the subproperty.

Remark You can also set the option in the same way with the [Macro definition] property in the [Frequently Used Options(for Assemble)] category on the [Common Options] tab.

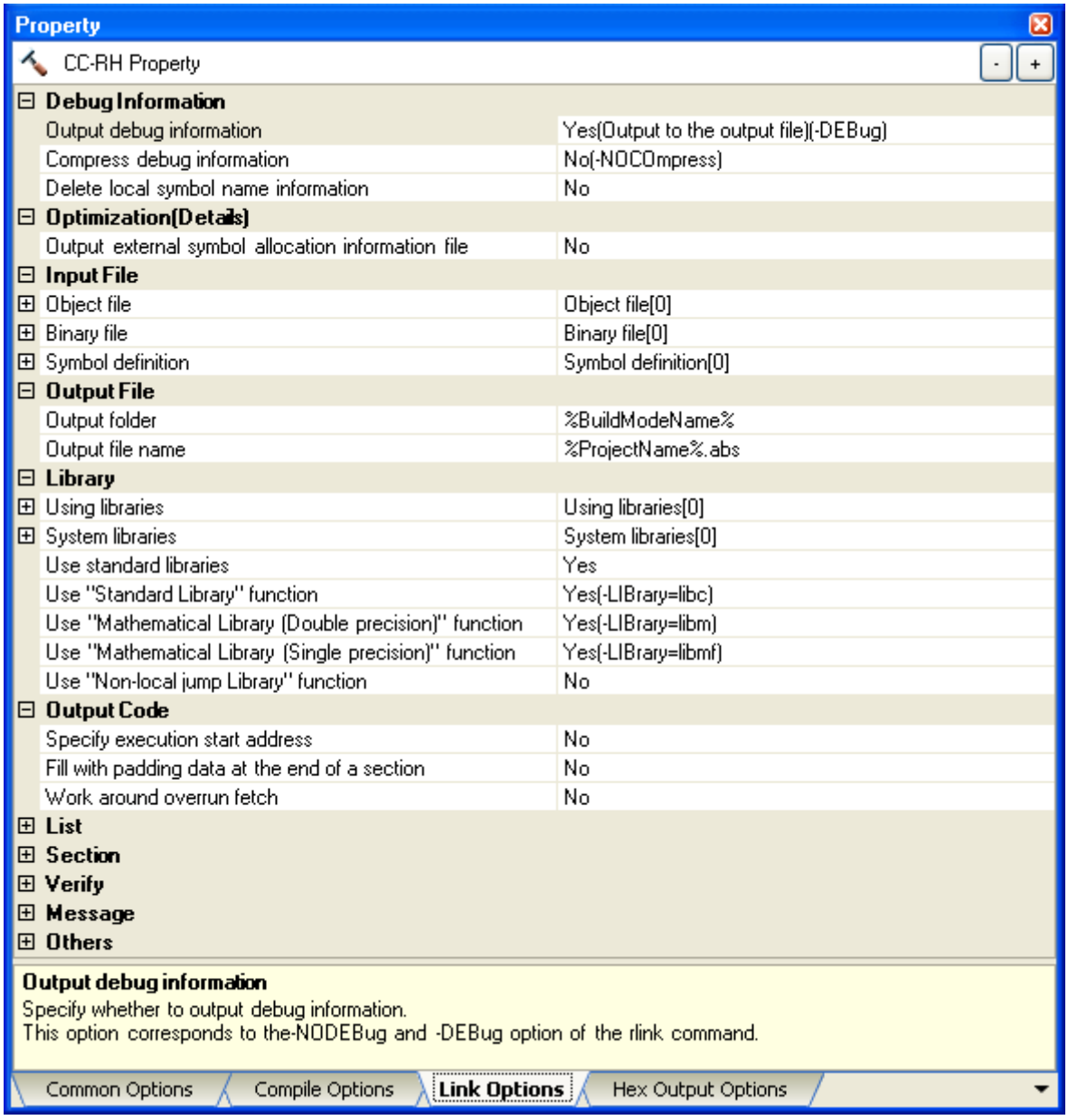
2.7 Set Link Options

To set options for the link phase, select the Build tool node on the project tree and select the [Link Options] tab on the Property panel.

You can set the various link options by setting the necessary properties in this tab.

Caution This tab is not displayed for the library project.

Figure 2-45. Property Panel: [Link Options] Tab

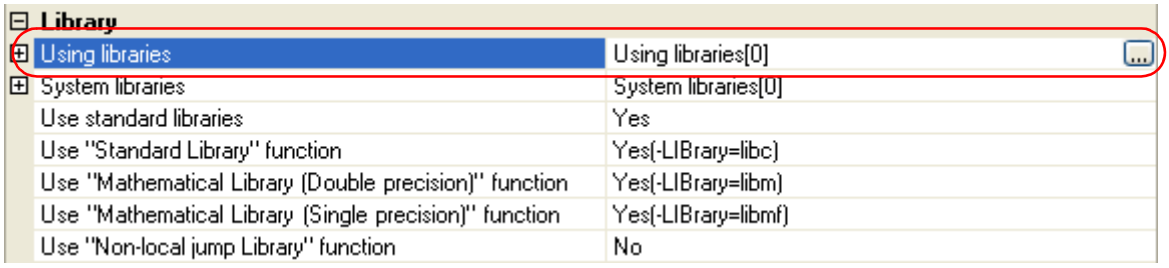


Remark Often used options have been gathered under the [Frequently Used Options(for Link)] category on the [Common Options] tab.

2.7.1 Add a user library

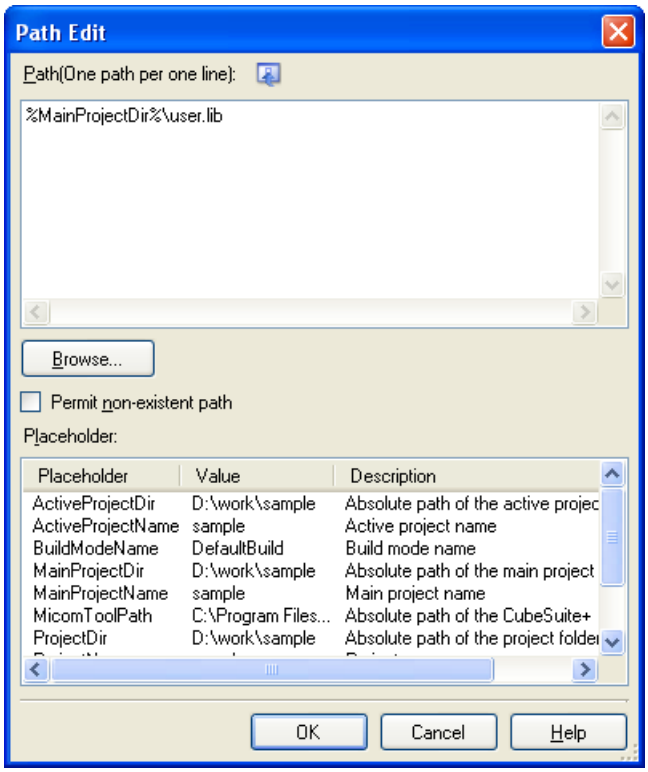
Select the build tool node on the project tree and select the [Link Options] tab on the Property panel.
Adding a user library is made with the [Using libraries] property in the [Library] category.

Figure 2-46. [Using libraries] Property



If you click the [...] button, the Path Edit dialog box will open.

Figure 2-47. Path Edit Dialog Box



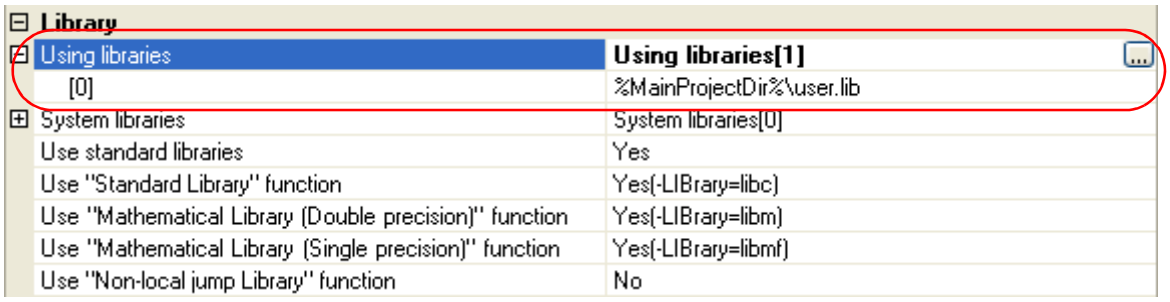
Enter the library file (including the path) per line in [Path(One path per one line)].
You can specify up to 259 characters per line, up to 65536 lines.

Remarks 1. This property supports placeholders.
If a line is double clicked in [Placeholder], the placeholder will be reflected in [Path(One path per one line)].

2. You can also specify the library file by one of the following procedures.
- Drag and drop the folder using such as Explorer.
- Click the [Browse...] button, and then select the folder in the Specify Using Library File dialog box.
- Double click a row in [Placeholder].

If you click the [OK] button, the entered library files are displayed as subproperties.

Figure 2-48. [Using libraries] Property (After Setting Library Files)



To change the library files, you can use the [...] button or enter the path directly in the text box of the subproperty.

Remark You can also set the option in the same way with the [Using libraries] property in the [Frequently Used Options(for Link)] category on the [Common Options] tab.

2.7.2 Prepare for using the overlaid section selection function

The optimizing linker (rlink) used by CC-RH can allocate multiple sections defined in a program to the same address. The sections allocated in this way are called "overlaid sections".

The debug tool provides a function to select the debug target section from the overlaid sections (priority sections) allocated to the same address. The function is called "overlaid section selection function".

A load module using overlaid sections can be debugged with switching of the priority section before program execution.

The method for generating a load module to use the overlaid section selection function is shown below.

(1) Copy the ROM area contents to RAM

Copy the ROM area contents to the RAM area to expand the code and data in the RAM.

(2) Set build options

Set the ROM-to-RAM mapped sections and overlaid sections to use the overlaid section selection function.

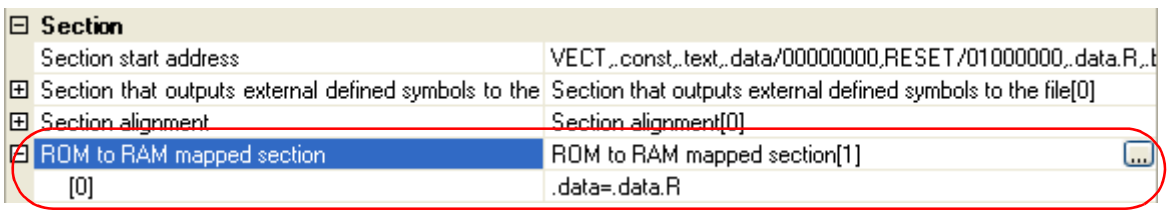
Select the build tool node on the project tree and select the [Link Options] tab on the Property panel.

(a) Set ROM-to-RAM mapped sections

Setting the ROM-to-RAM mapped sections is made with the [ROM to RAM mapped section] property in the [Section] category.

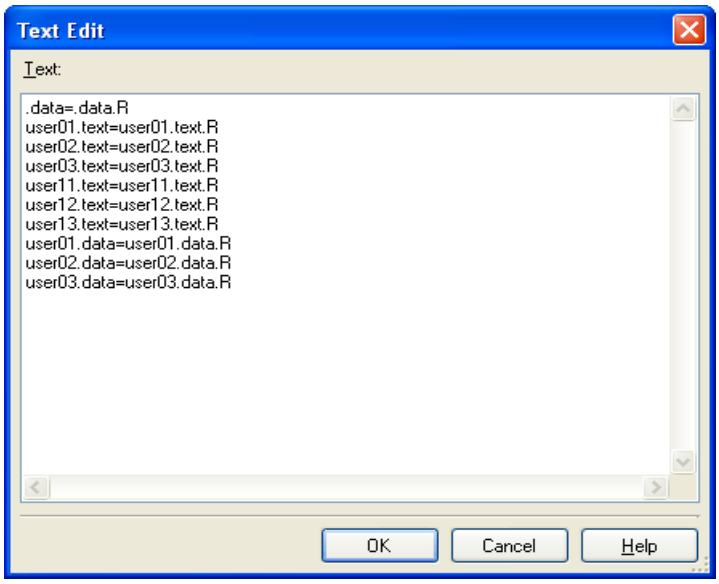
This reserves the RAM section with the same size as that of the ROM section and relocates the symbols defined in the ROM section to addresses in the RAM section.

Figure 2-49. [ROM to RAM mapped section] Property



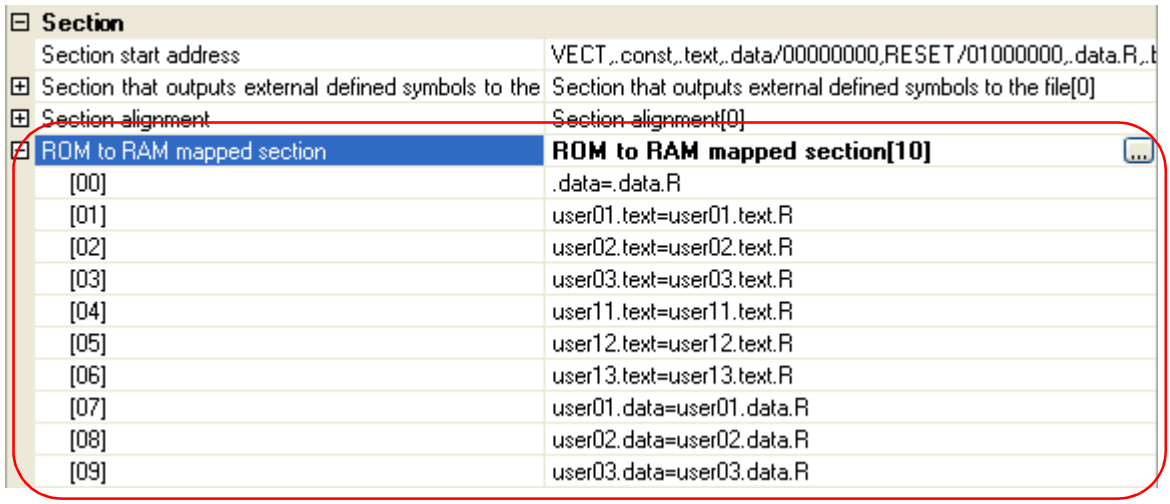
If you click the [...] button, the Text Edit dialog box will open.

Figure 2-50. Text Edit Dialog Box



Enter the section name in [Text] in the format of "*ROM section name=RAM section name*", with one section name per line.
You can specify up to 32767 characters per line, up to 65535 lines.
If you click the [OK] button, the entered section names are displayed as subproperties.

Figure 2-51. [ROM to RAM mapped section] Property (After Setting Sections)

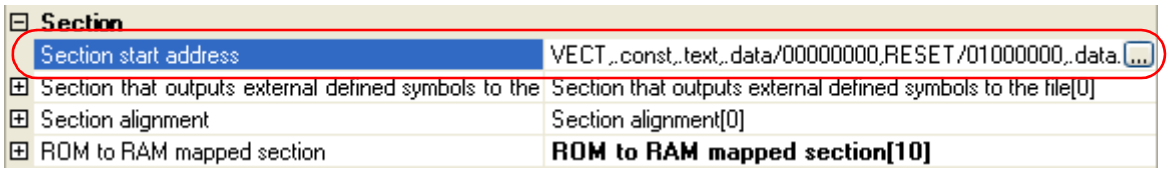


To change the section names, you can use the [...] button or enter them directly in the text box of the subproperty.

(b) Set ROM sections and RAM sections (overlaid sections)

Setting the sections is made with the [Section start address] property in the [Section] category.

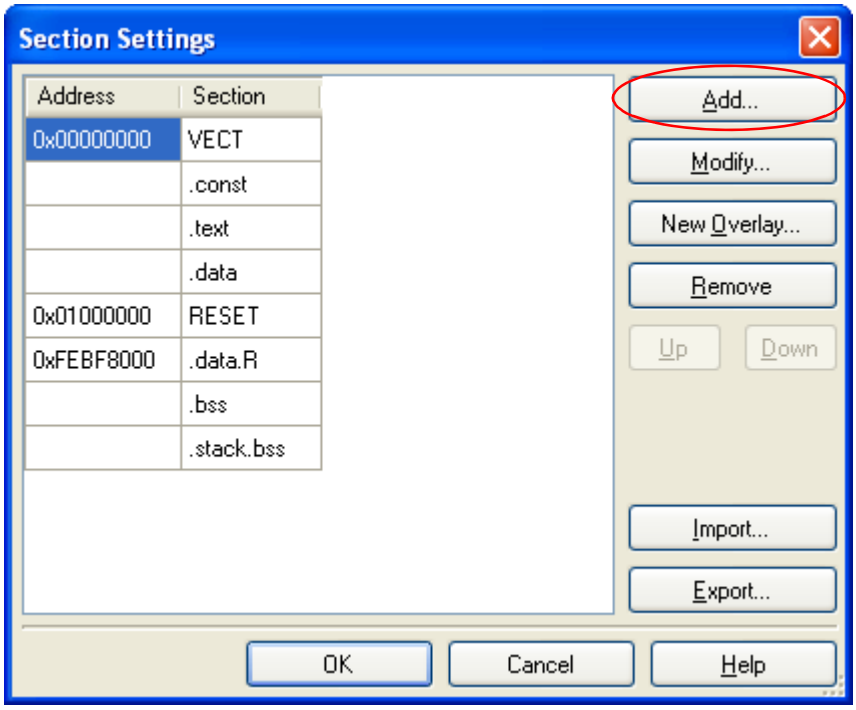
Figure 2-52. [Section start address] Property



<1> Set ROM sections

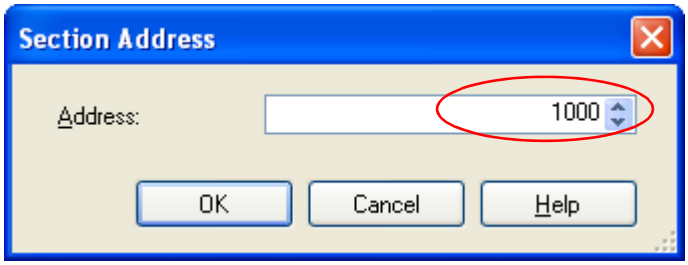
If you click the [...] button, the Section Settings dialog box will open.

Figure 2-53. Section Settings Dialog Box



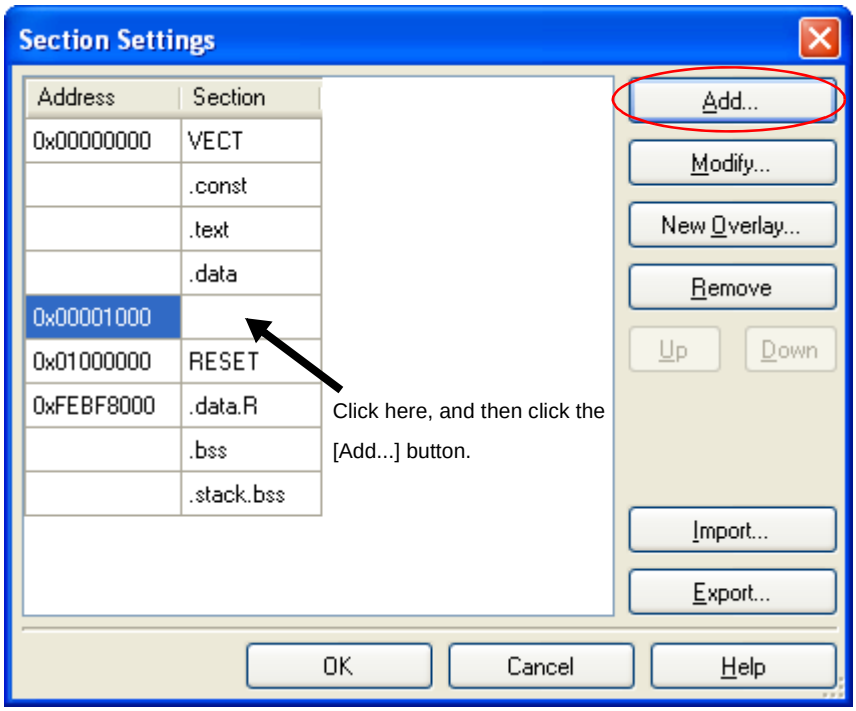
If you click the [Add...] button, the Section Address dialog box will open.

Figure 2-54. Section Address Dialog Box



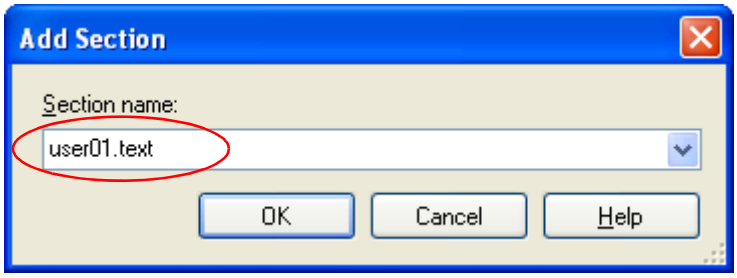
Enter in [Address] the address of the ROM section to be added and click the [OK] button to add the entered address to [Address] in the Section Settings dialog box.

Figure 2-55. Section Settings Dialog Box (After ROM Section Addresses Are Added)



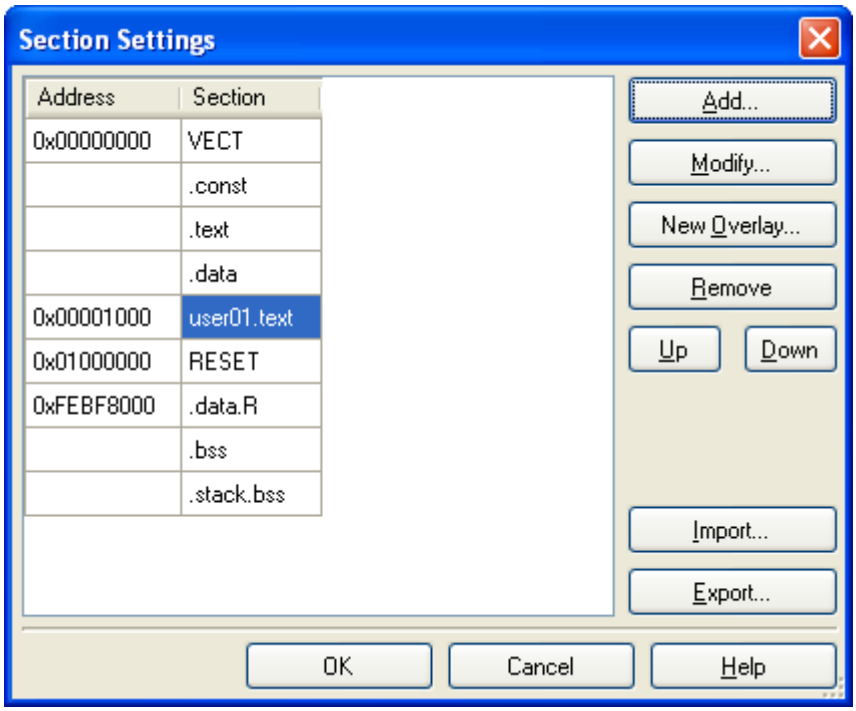
Click the Section column on the added address row and click the [Add...] button to open the Add Section dialog box.

Figure 2-56. Add Section Dialog Box



Enter in [Section name] the name of the ROM section to be added and click the [OK] button to add the entered section to [Section] in the Section Settings dialog box.

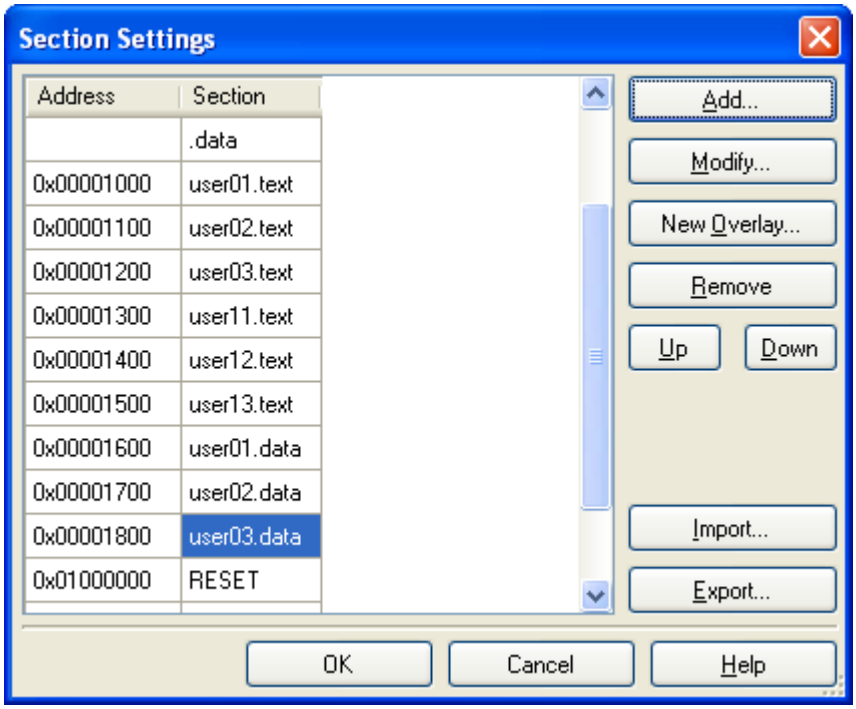
Figure 2-57. Section Settings Dialog Box (After ROM Sections Are Added)



For other ROM sections, set addresses and section names in the same way.

Remark Click the Address column and click the [Add...] button to open the Section Address dialog box, allowing you to add a new address.

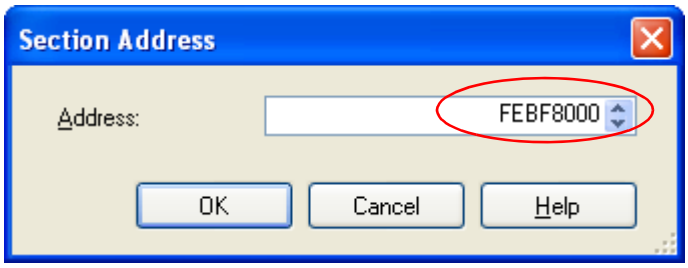
Figure 2-58. Section Settings Dialog Box (After Multiple ROM Sections Are Added)



<2> Set RAM sections (overlaid sections)

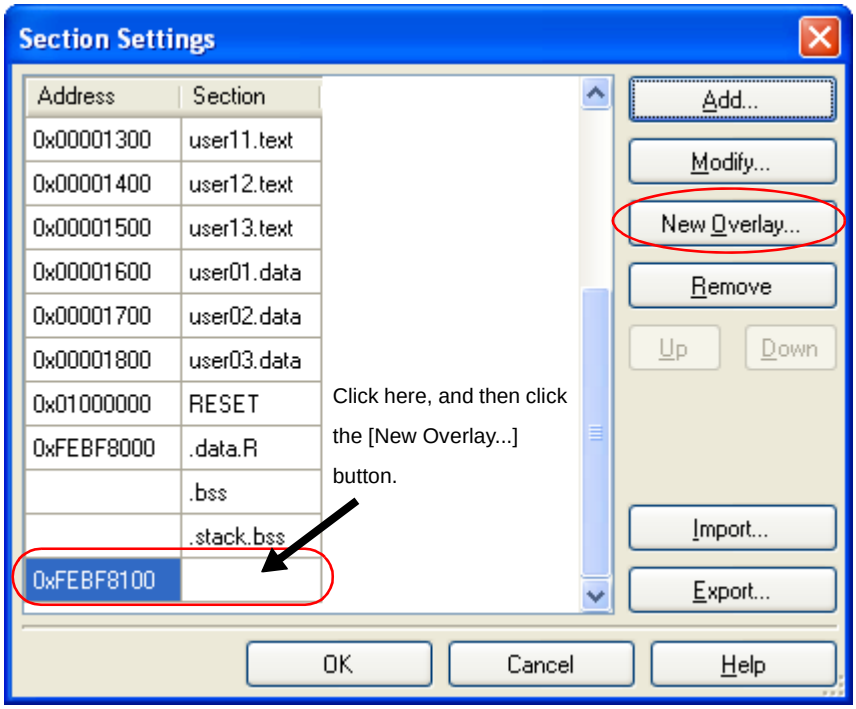
Click an added address and click the [Add...] button to open the Section Address dialog box.

Figure 2-59. Section Address Dialog Box



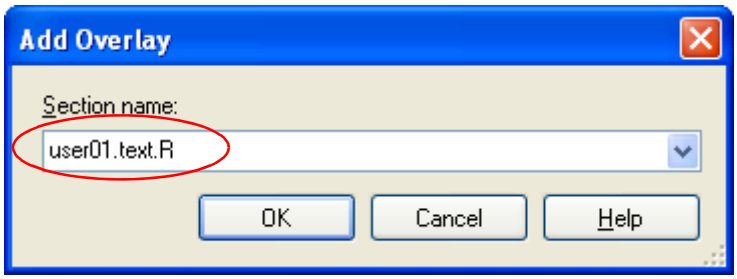
Enter in [Address] the address of the RAM section to be added and click the [OK] button to add the entered address to [Address] in the Section Settings dialog box.

Figure 2-60. Section Settings Dialog Box (After RAM Section Addresses Are Added)



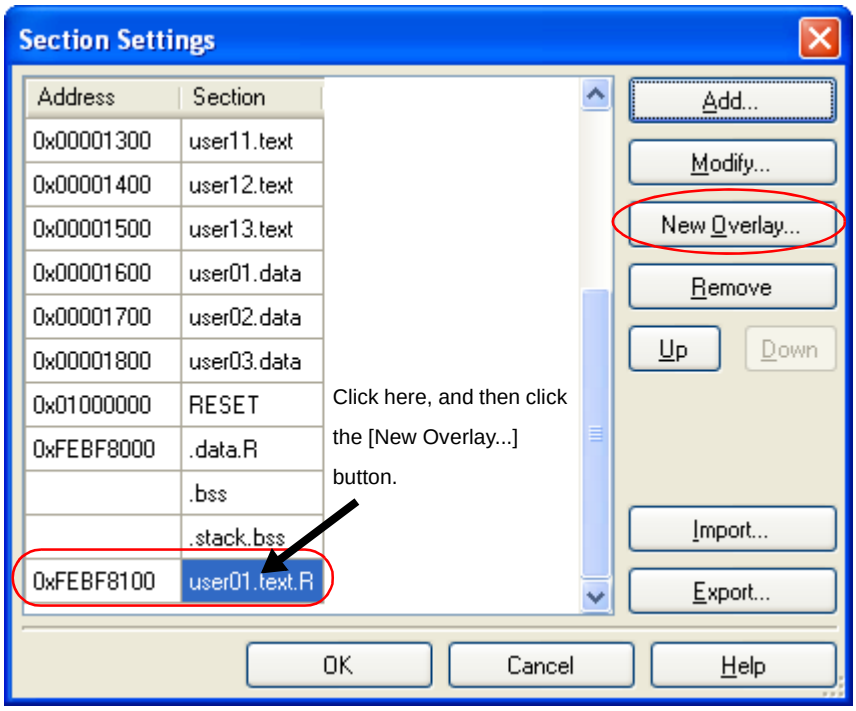
Click the added address row (Address column or Section column) and click the [New Overlay...] button to open the Add Overlay dialog box.

Figure 2-61. Add Overlay Dialog Box



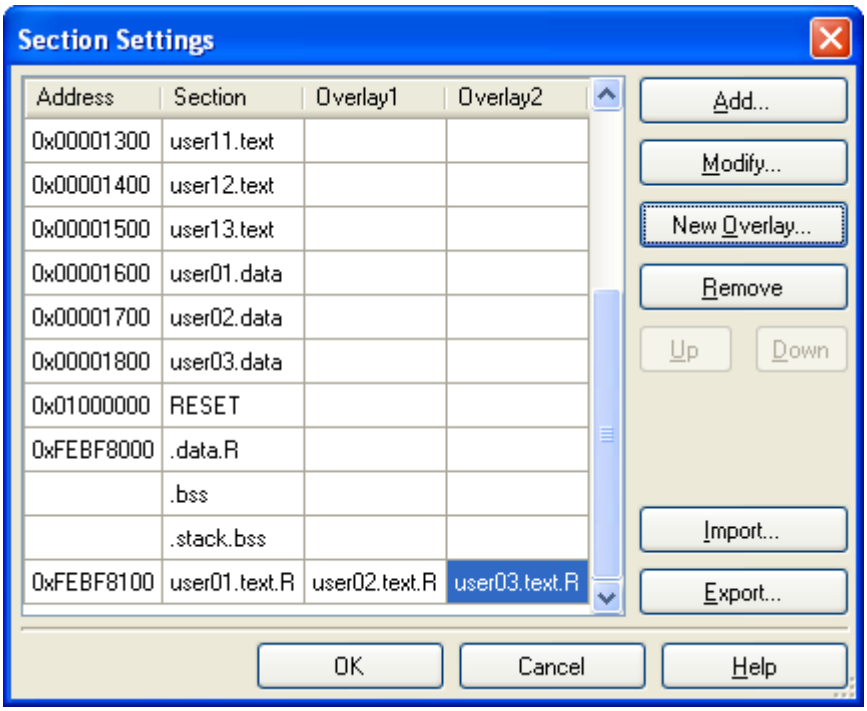
Enter in [Section name] the name of the RAM section to be added and click the [OK] button to add the entered section to [Section] in the Section Settings dialog box.

Figure 2-62. Section Settings Dialog Box (After RAM Sections Are Added)



Add the sections to be allocated to the same address by using the [New Overlay...] button in the same way.
The added sections are displayed under [Overlay *n*] (*n*: number starting with "1").

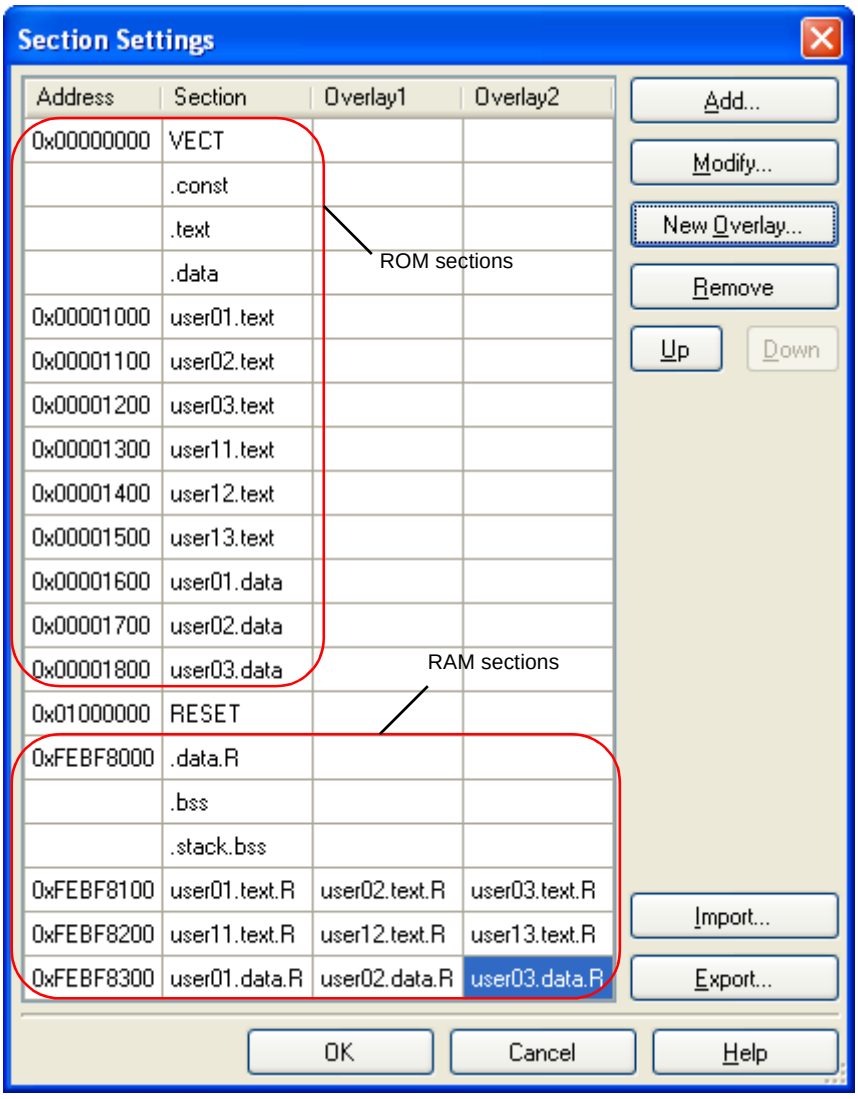
Figure 2-63. Section Settings Dialog Box (After Overlaid Sections Are Added)



For other RAM sections, set addresses and section names in the same way.

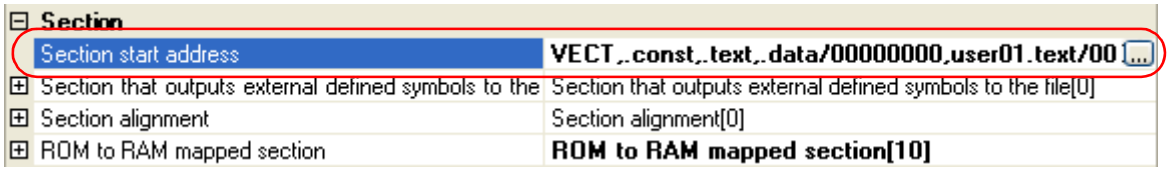
Remark Click the Address column and click the [Add...] button to open the Section Address dialog box, allowing you to add a new address.

Figure 2-64. Section Settings Dialog Box (After Multiple RAM Sections Are Added)



Click the [OK] button. The specified ROM sections and RAM sections (overlaid sections) will be displayed in the text boxes.

Figure 2-65. [Section start address] Property (After Setting Sections)



(3) Run a build of the project

Run a build of the project.

A load module file to use the overlaid section selection function is generated.

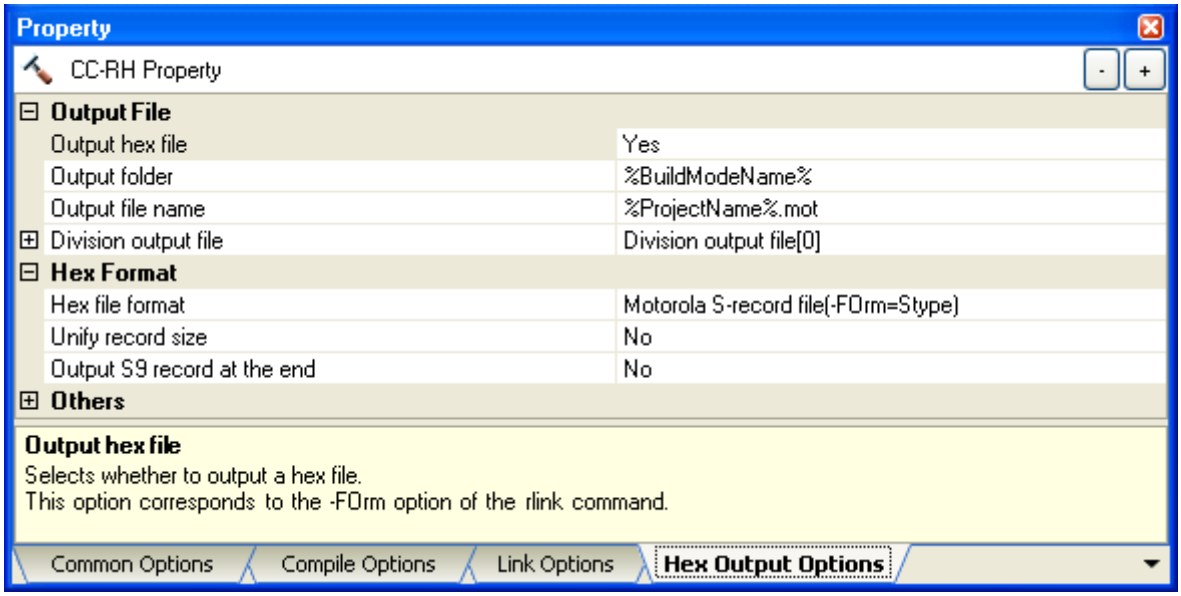
2.8 Set Hex Output Options

To set options for the hex output phase, select the Build tool node on the project tree and select the [Hex Output Options] tab on the Property panel.

You can set the various hex output options by setting the necessary properties in this tab.

Caution This tab is not displayed for the library project.

Figure 2-66. Property Panel: [Hex Output Options] Tab



Remark Often used options have been gathered under the [Frequently Used Options(for Hex Output)] category on the [Common Options] tab.

2.8.1 Set the output of a hex file

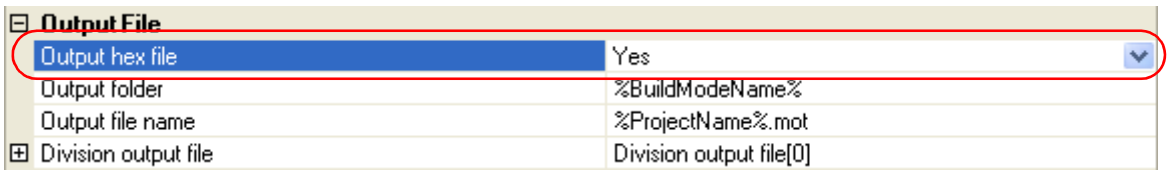
Select the build tool node on the project tree and select the [Hex Output Options] tab on the Property panel.

(1) Set the output of a hex file

The setting to output a hex file is made with the [Output hex file] property in the [Output File] category.

To output a hex file, select [Yes], to not output a hex file, select [No].

Figure 2-67. [Output hex file] Property



When outputting a hex file, you can set the output folder and output file name.

(a) Set the output folder

Setting the output folder is made with the [Output folder] property by directly entering to the text box or by the [...] button.

Up to 247 characters can be specified in the text box.

This property supports the following placeholder.

%ActiveProjectDir%: Replaces with the absolute path of the active project folder.

%ActiveProjectName%: Replaces with the active project name.

%BuildModeName%: Replaces with the build mode name.

%MainProjectDir%: Replaces with the absolute path of the main project folder.

%MainProjectName%: Replaces with the main project name.

%MicomToolPath%: Replaces with the absolute path of the install folder of this product.

%ProjectDir%: Replaces with the absolute path of the project folder.

%ProjectName%: Replaces with the project name.

%TempDir%: Replaces with the absolute path of the temporary folder.

%WinDir%: Replaces with the absolute path of the Windows system folder.

"%BuildModeName%" is set by default.

(b) Set the output file name

Setting the output file is made with the [Output file name] property by directly entering to the text box.

Up to 259 characters can be specified in the text box.

This property supports the following placeholders.

%ActiveProjectName%: Replaces with the active project name.

%MainProjectName%: Replaces with the main project name.

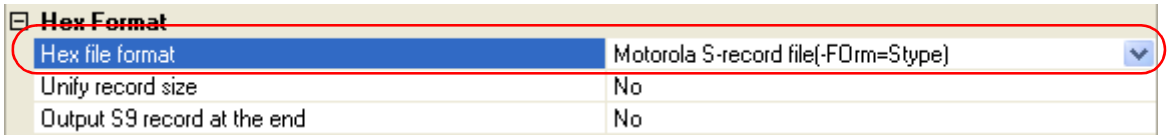
%ProjectName%: Replaces with the project name.

"%ProjectName%.mot" is set by default.

(2) Set the hex file format

Select the format in the [Hex file format] property in the [Hex Format] category.

Figure 2-68. [Hex file format] Property



You can select any of the formats below.

Format	Configuration
Intel HEX file(-FOrm=Hexadecimal)	Outputs an Intel HEX file.
Motorola S-record file(-FOrm=Stype)	Outputs a Motorola S-record file.
Binary file(-FOrm=Binary)	Outputs a binary file.

Remark See "3.5 Intel HEX File" and "3.6 Motorola S-record File" for details about the Intel Hex file and Motorola S-record file.

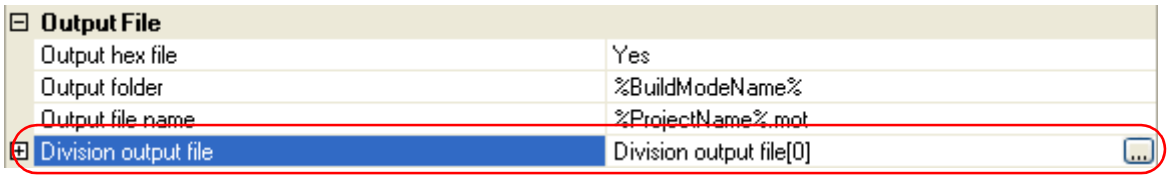
2.8.2 Fill the vacant area

Select the build tool node on the project tree and select the [Hex Output Options] tab on the Property panel.

(1) Set the hex file output range

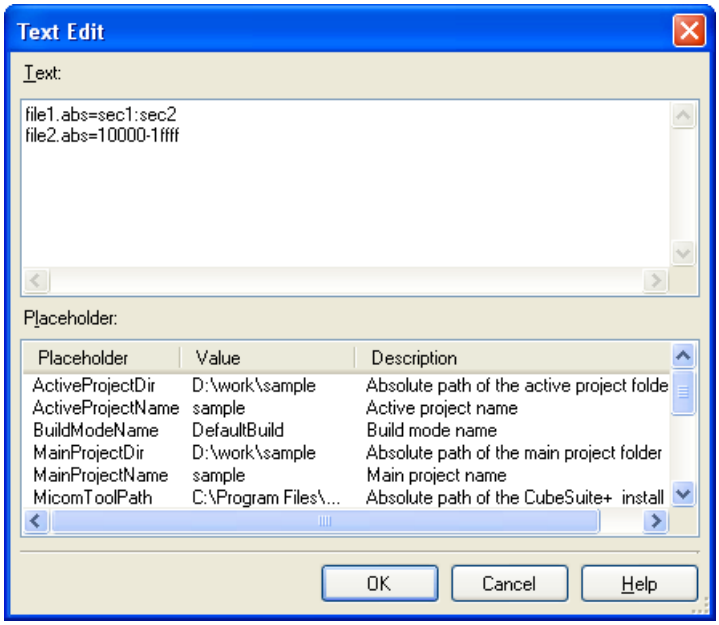
The setting of the hex file output range is made with the [Division output file] property in the [Output File] category.

Figure 2-69. [Division output file] Property



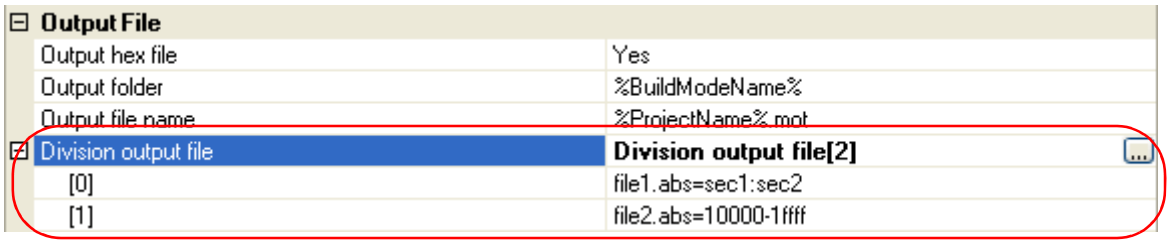
If you click the [...] button, the Text Edit dialog box will open.

Figure 2-70. Text Edit Dialog Box



Specify the division output file name in [Text] in the format of "*file name=start address-end address*" (*start address*, *end address*: The start address and end address of the output range) or "*file name=section name*" (*section name*: The name of the output section), with one file name per line.
If multiple section names are specified, delimit them with a colon as in "*file name=section name*:*section name*".
Specify the start address and end address in hexadecimal.
You can specify up to 259 characters per line, up to 65535 lines.
If you click the [OK] button, the entered division output file names are displayed as subproperties.

Figure 2-71. [Division output file] Property (After Setting Division Output File Names)



To change the division output file names, you can use the [...] button or enter them directly in the text box of the subproperty.

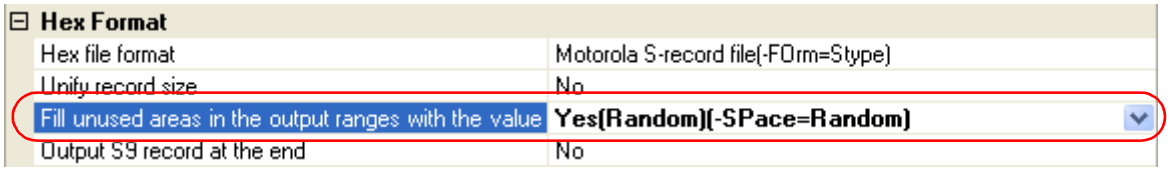
(2) Set the method for filling the vacant area

Set the method for filling the vacant area in the output range.

(a) Fill the vacant area with random numbers

Select [Yes(Random)(-SPace=Random)] in the [Fill unused areas in the output ranges with the value] property in the [Hex Format] category.

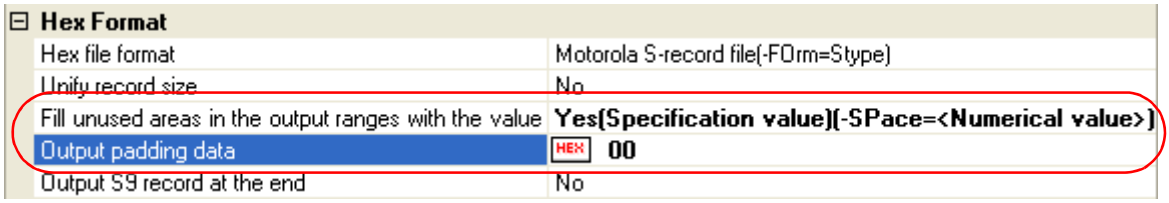
Figure 2-72. [Fill unused areas in the output ranges with the value] Property



(b) Specify data to fill the vacant area

Select [Yes(Specification value)(-SPace=<Numerical value>)] in the [Fill unused areas in the output ranges with the value] property in the [Hex Format] category. The [Output padding data] property will be displayed.

Figure 2-73. [Fill unused areas in the output ranges with the value] and [Output padding data] Property



Enter the fill value for the vacant area directly in the text box.
The range that can be specified for the value is 00 to FFFFFFFFFF (hexadecimal number).
"FF" is set by default.

2.8.3 Combine multiple objects

For a multi-core project, the Intel HEX files or Motorola S-record files in the projects specified as constituent applications can be combined into a single hex file (This is called the object combine function).

Note that Intel HEX files and Motorola S-record files cannot be mixed.

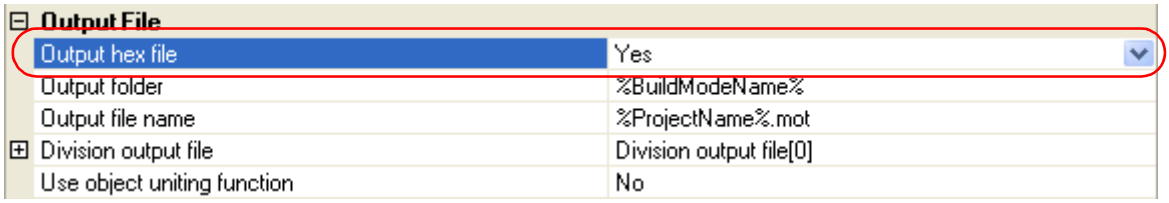
(1) Set the output of a hex file

Set the output of a hex file and the hex file format in the boot loader project and each application project.

Select the build tool node on the project tree and select the [Hex Output Options] tab on the Property panel.

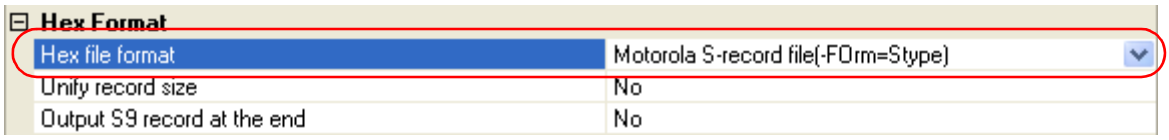
Select [Yes] in the [Output hex file] property in the [Output File] category.

Figure 2-74. [Output hex file] Property



Select [Intel HEX file(-FOrm=Hexadecimal)] or [Motorola S-record file(-FOrm=Stype)] in the [Hex file format] property in the [Hex Format] category.

Figure 2-75. [Hex file format] Property



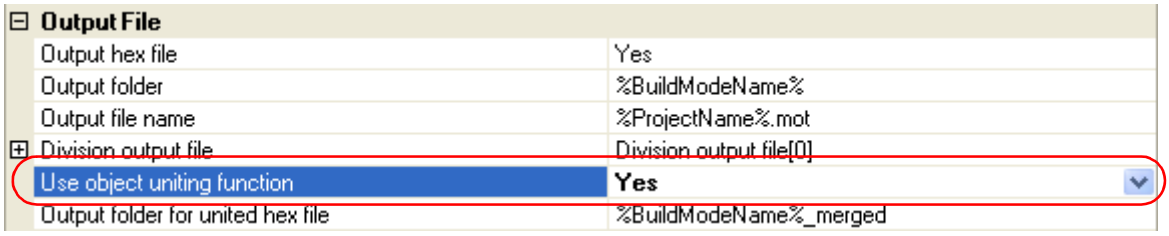
Caution The same format should be selected in the boot loader project and all application projects.

(2) Specify combining into a hex file

Specify combining into a hex file in the boot loader project.

Select [Yes] in the [Use object uniting function] property in the [Output File] category.

Figure 2-76. [Use object uniting function] Property



The destination where the combined hex file is to be output can be specified in [Output folder for united hex file] property (default: %BuildModeName%_merged).

The name in the [Output file name] property is used as the combined file name.

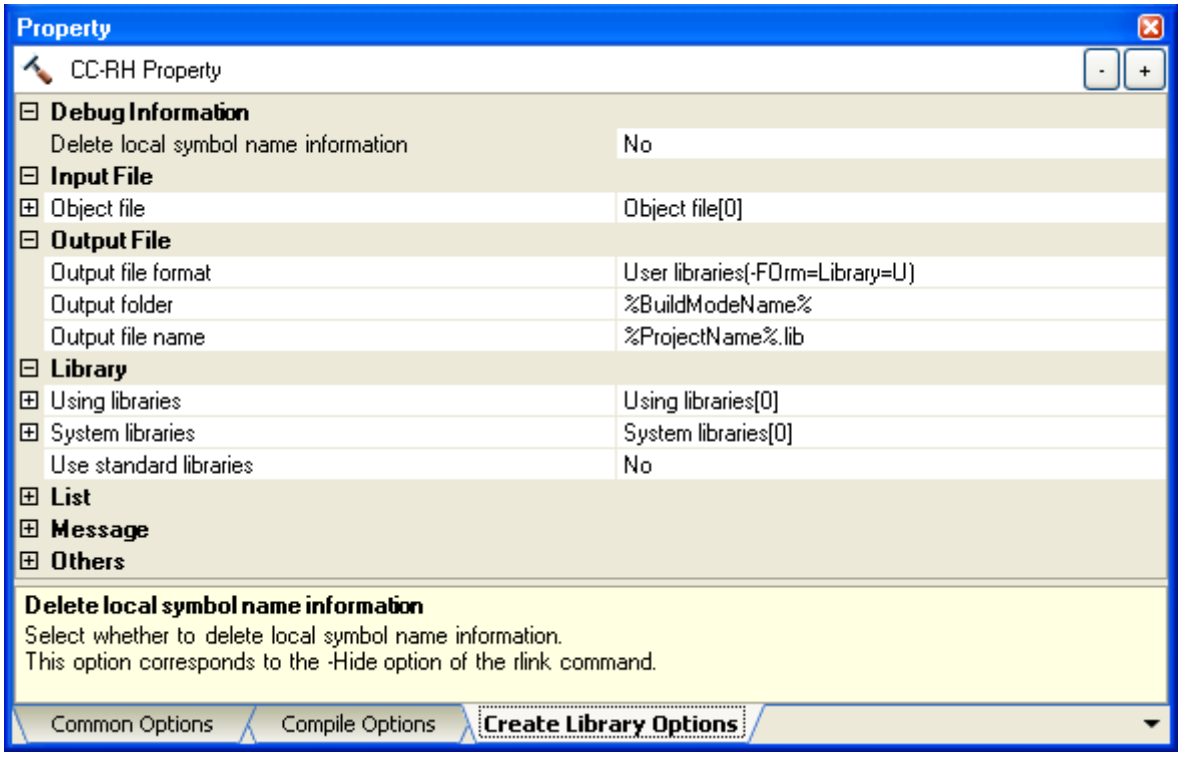
2.9 Set Create Library Options

To set options for the librarian, select the Build tool node on the project tree and select the [Create Library Options] tab on the Property panel.

You can set the various create library options by setting the necessary properties in this tab.

Caution This tab is displayed for the library project.

Figure 2-77. Property Panel: [Create Library Options] Tab



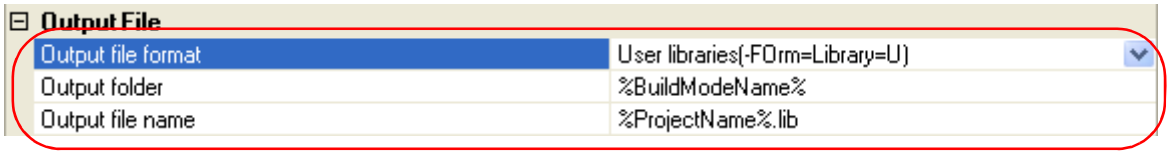
Remark Often used options have been gathered under the [Frequently Used Options(for Create Library)] category on the [Common Options] tab.

2.9.1 Set the output of a library file

Select the build tool node on the project tree and select the [Create Library Options] tab on the Property panel.

The setting to output a library file is made with the [Output File] category.

Figure 2-78. [Output File] Category



(1) Set the output format

Select the format in the [Output file format] property.

You can select any of the formats below.

Format	Configuration
User libraries(-FOrm=Library=U)	Outputs a user library file.
System libraries(-FOrm=Library=S)	Outputs a system library file. The system library file is linked after the user library file. Select this item to create a library that is to be linked after the user library file.
Relocatable file(-FOrm=Relocate)	Outputs a relocatable file.

(2) Set the output folder

Setting the output folder is made with the [Output folder] property by directly entering to the text box or by the [...] button.

Up to 247 characters can be specified in the text box.

This property supports the following placeholder.

%ActiveProjectDir%: Replaces with the absolute path of the active project folder.

%ActiveProjectName%: Replaces with the active project name.

%BuildModeName%: Replaces with the build mode name.

%MainProjectDir%: Replaces with the absolute path of the main project folder.

%MainProjectName%: Replaces with the main project name.

%MicomToolPath%: Replaces with the absolute path of the install folder of this product.

%ProjectDir%: Replaces with the absolute path of the project folder.

%ProjectName%: Replaces with the project name.

%TempDir%: Replaces with the absolute path of the temporary folder.

%WinDir%: Replaces with the absolute path of the Windows system folder.

"%BuildModeName%" is set by default.

(3) Set the output file name

Setting the output file is made with the [Output file name] property by directly entering to the text box.

If the extension is omitted, it is automatically added according to the selection in the [Output file format] property.

When [User libraries(-FOrm=Library=U)] is selected: .lib

When [System libraries(-FOrm=Library=S)] is selected: .lib

When [Relocatable file(-FOrm=Relocate)] is selected: .rel

Up to 259 characters can be specified in the text box.

This property supports the following placeholders.

%ActiveProjectName%: Replaces with the active project name.

%MainProjectName%: Replaces with the main project name.

%ProjectName%: Replaces with the project name.

"%ProjectName%.lib" is set by default.

2.10 Set Build Options Separately

Build options are set at the project or file level.

Project level: See "2.10.1 Set build options at the project level"

File level: See "2.10.2 Set build options at the file level"

2.10.1 Set build options at the project level

To set options for build options for the project (main project or subproject), select the Build tool node on the project tree to display the Property panel.

Select the phase tab and set build options by setting the necessary properties.

Compile phase: [Compile Options] tab

Assemble phase: [Assemble Options] tab

Link phase: [Link Options] tab

Hex output phase: [Hex Output Options] tab

Create library phase: [Create Library Options] tab

2.10.2 Set build options at the file level

You can individually set compile and assemble options for each source file added to the project.

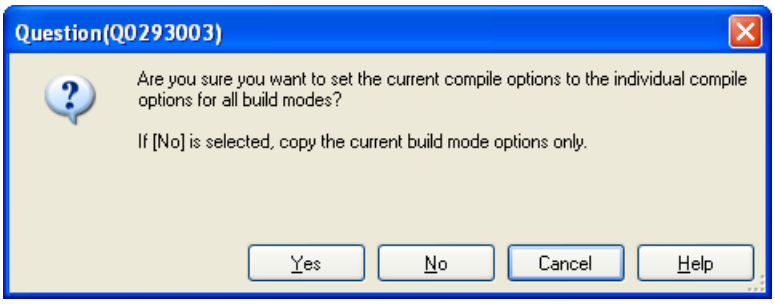
(1) When setting compile options for a C source file

Select the C source file on the project tree and select the [Build Settings] tab on the Property panel.

Select [Yes] in the [Set individual compile option] property in the [Build] category. The Message Dialog Box will open.

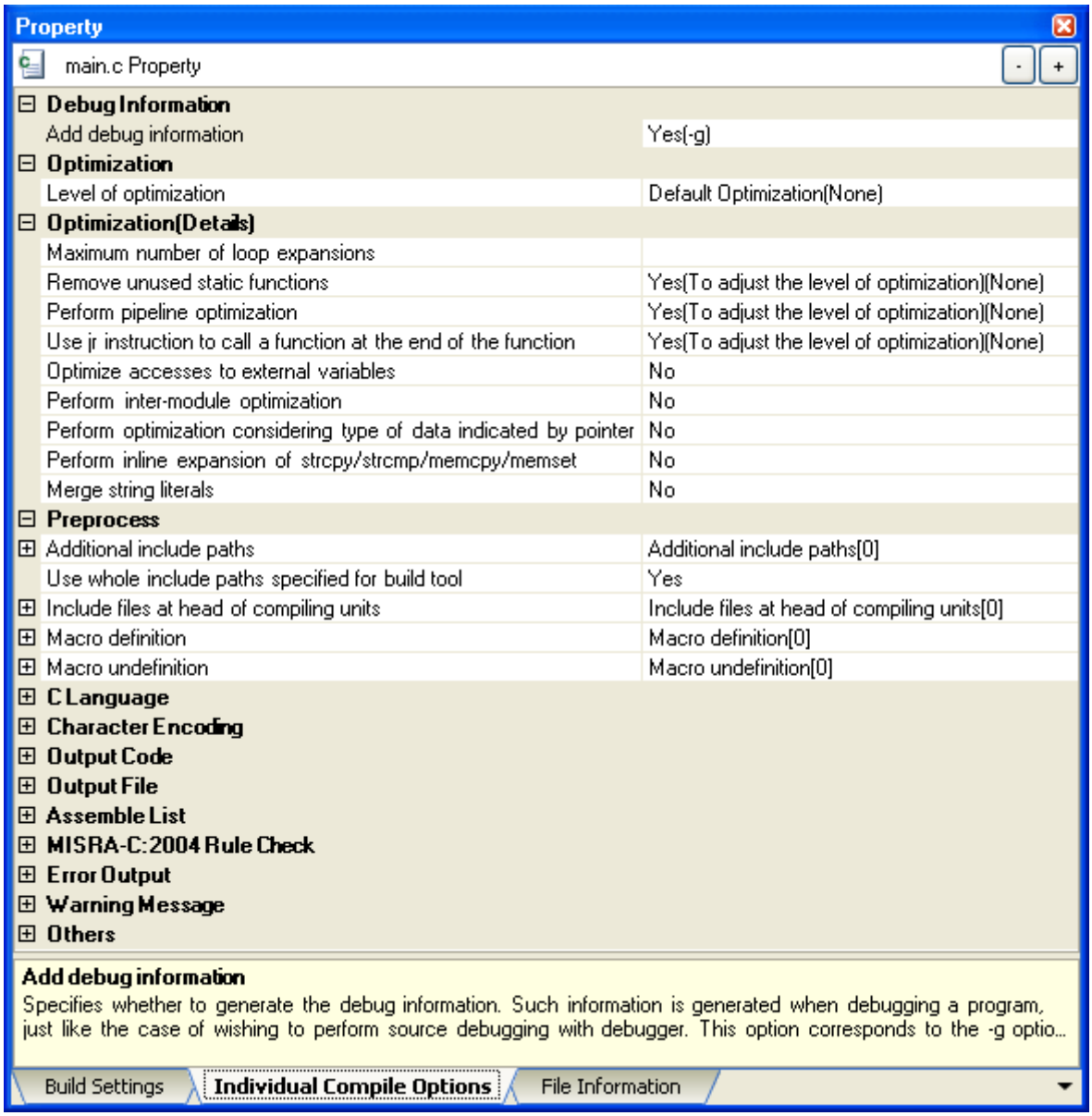
Figure 2-79. [Set individual compile option] Property

Figure 2-80. Message Dialog Box



Click [Yes] in the dialog box. The [Individual Compile Options] tab will be displayed.

Figure 2-81. Property Panel: [Individual Compile Options] Tab



You can set compile options for the C source file by setting the necessary properties in this tab.

Note that this tab takes over the settings of the [Common Options] tab and [Compile Options] tab by default except the properties shown below.

- [Additional include paths] and [Use whole include paths specified for build tool] in the [Preprocess] category
- [Object module file name] in the [Output File] category

(2) When setting assemble options for an assembly source file

Select the assembly source file on the project tree and select the [Build Settings] tab on the Property panel. Select [Yes] in the [Set individual assemble option] property in the [Build] category. The Message Dialog Box (When A Build Is Run Simultaneously) or Message Dialog Box (When A Build Is Not Run Simultaneously) will open.

Figure 2-82. [Set individual assemble option] Property

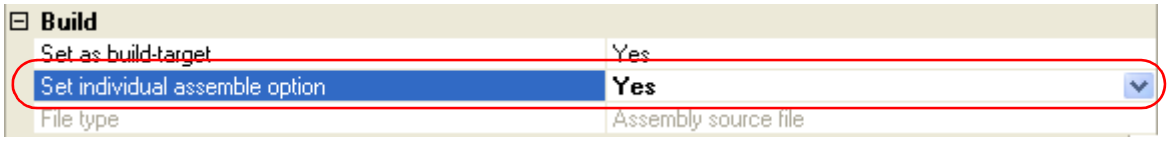


Figure 2-83. Message Dialog Box (When A Build Is Run Simultaneously)

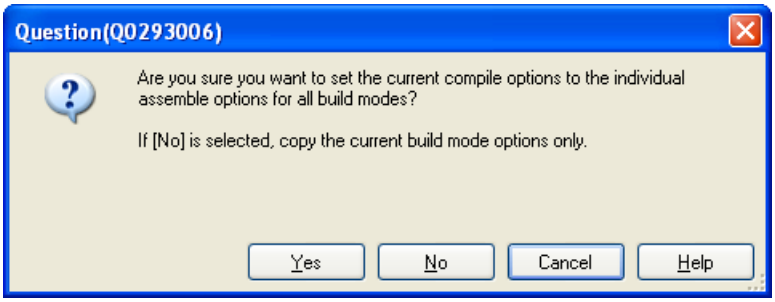
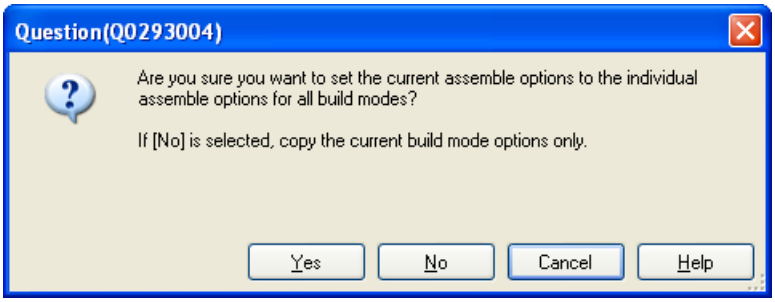
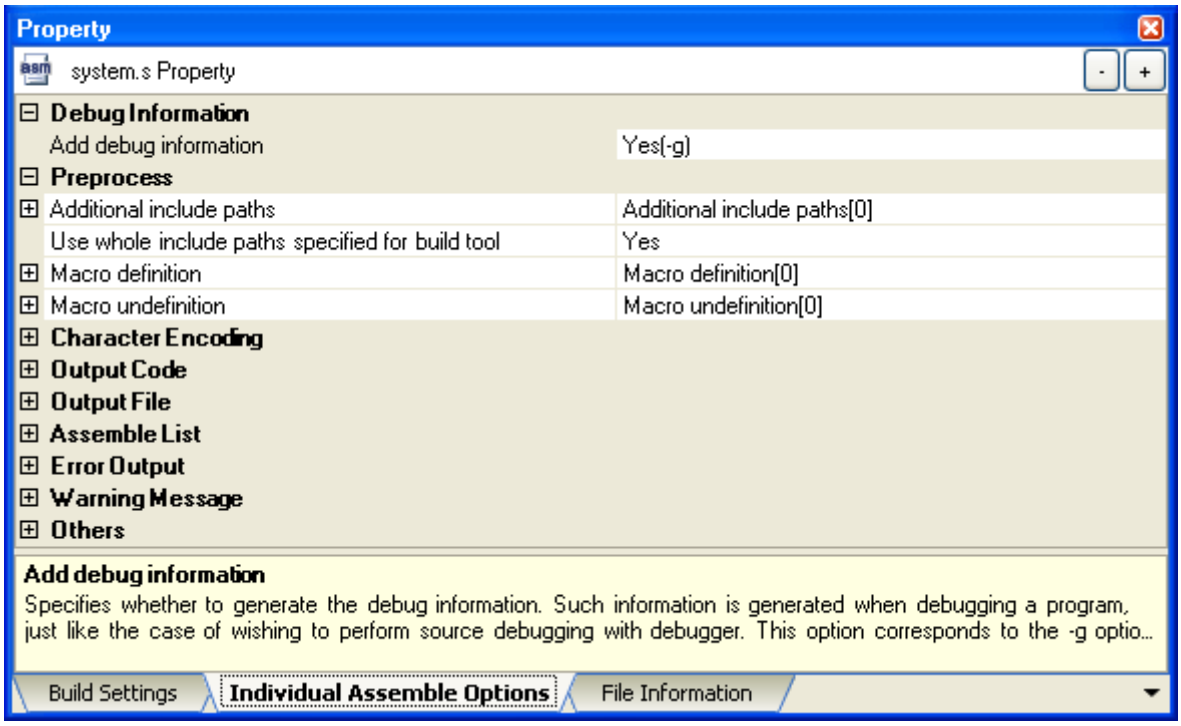


Figure 2-84. Message Dialog Box (When A Build Is Not Run Simultaneously)



Click [Yes] in the dialog box. The [Individual Assemble Options] tab will be displayed.

Figure 2-85. Property Panel: [Individual Assemble Options] Tab



You can set assemble options for the assembly source file by setting the necessary properties in this tab.

Note that this tab takes over the settings of the [Common Options] tab and [Compile Options] tab/[Assemble Options] tab by default except the properties shown below.

- [Additional include paths] and [Use whole include paths specified for build tool] in the [Preprocess] category
- [Object module file name] in the [Output File] category

2.11 Make Settings for Build Operations

This section explains operations on a build.

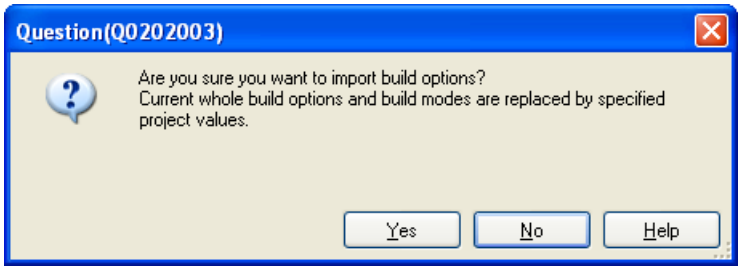
- Import the build options of other project
- Set the link order of files
- Change the build order of subprojects
- Display a list of build options
- Change the build target project
- Add a build mode
- Change the build mode
- Delete a build mode
- Set the current build options as the standard for the project

2.11.1 Import the build options of other project

You can import the build options of other project to the current project.

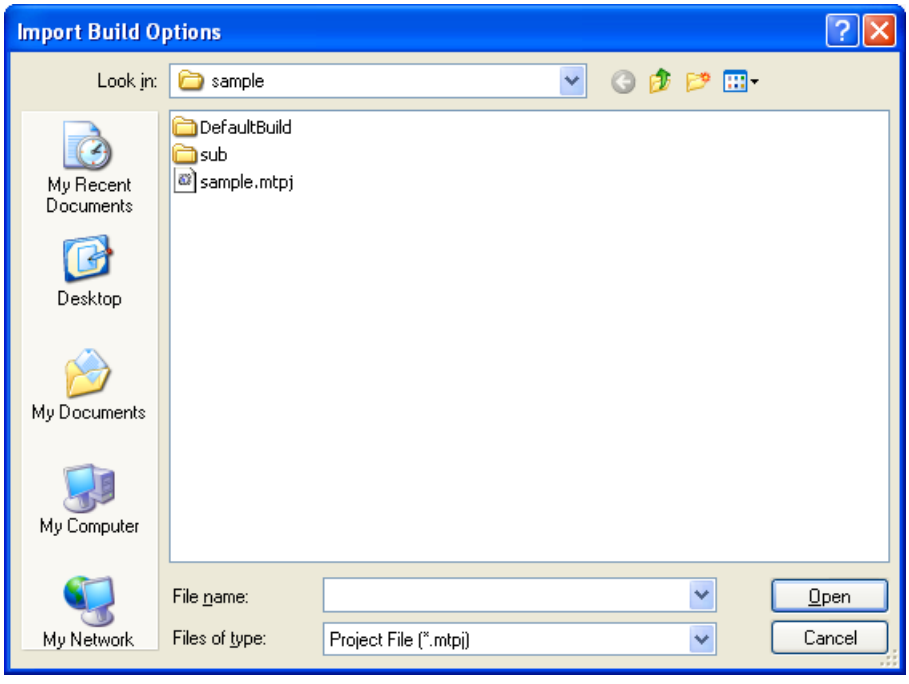
On the project tree, select the Build tool node, and then select [Import Build Options...] from the context menu. The following message dialog box will open.

Figure 2-86. Message Dialog Box



Click [Yes] in the dialog box. The Import Build Options dialog box will open.

Figure 2-87. Import Build Options Dialog Box



In the dialog box, select the target project file for import the build options and click the [Open] button.
The build options of the selected project file are imported to the current project.

Remarks 1. The conditions of the project that is importable are shown below.
- The build tool is the same.
- The type of the project (application, library, etc.) is the same.
- The project has been created by CubeSuite+ with the same version.

2. The target build options for importing are only the general options set in the properties of the build tool.
The setting of the standard build options (see "2.11.9 Set the current build options as the standard for the project") and the individual options are not imported.

3. All the build modes of the import target are imported.
However, the build modes of the current project (other than DefaultBuild) are deleted.

4. The version of the build tool to be used is imported.

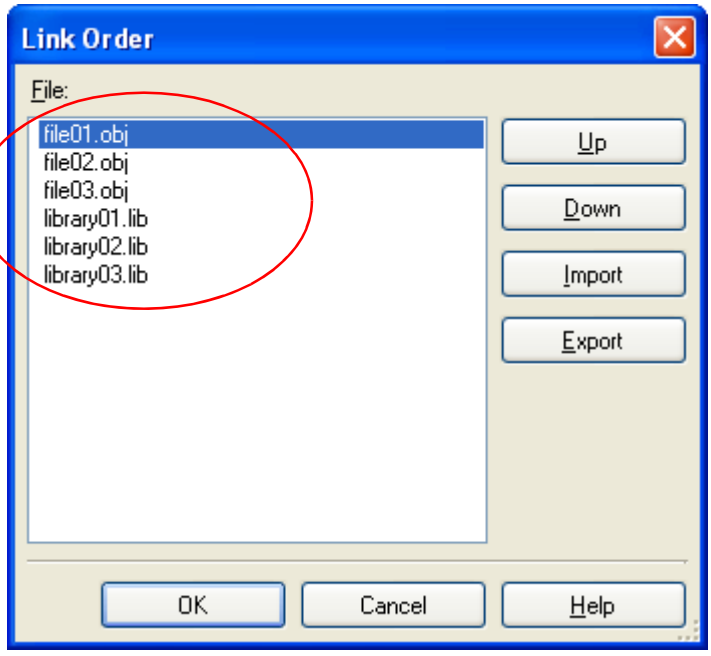
2.11.2 Set the link order of files

The link order of object module files and library files is decided automatically, but you can also set the order.
The procedures for performing this operation are described below.

(1) Open the Link Order dialog box

On the project tree, select the Build tool node, and then select [Set Link Order...] from the context menu. The Link Order dialog box will open.

Figure 2-88. Link Order Dialog Box



The names of the following files are listed in [File] in the order that the files are input to the linker.
- Object module files which are generated from the source files added to the selected main project or subproject
- Object module files which are added directly to the project tree of the selected main project or subproject
- Library files which are added directly to the project tree of the selected main project or subproject

Remark The default order is the order that the files are added to the project.
Object module files generated from newly added source files and newly added object module files are added after the last object module file in the list.
Newly added library files are added to the end of the list.

(2) Change the file display order

By changing the display order of the files, you can set the input order of the files to the linker.
Change the file display order by one of the following procedures.

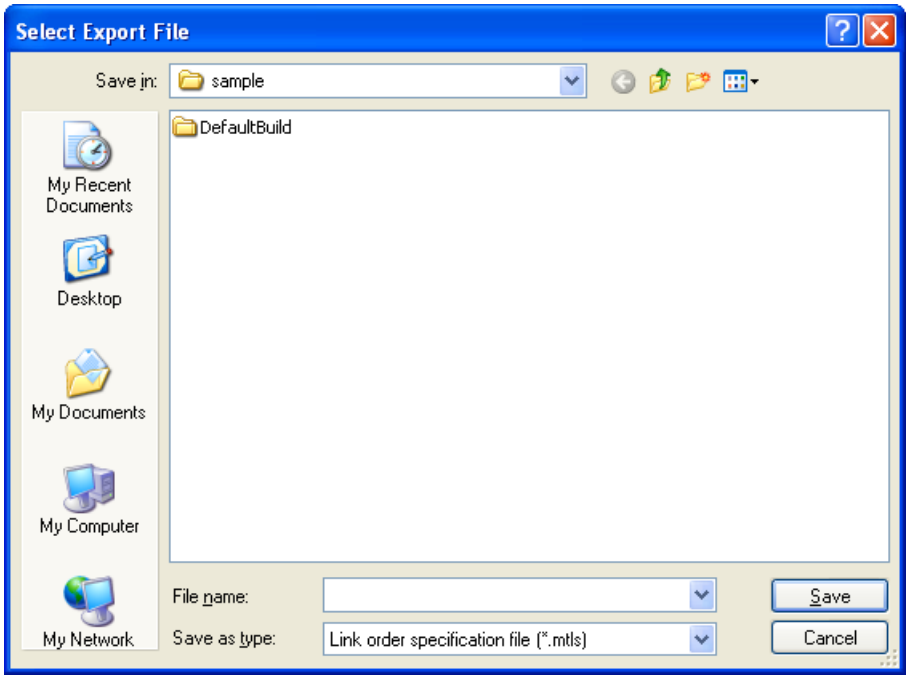
- Move the file name by using the [Up] and [Down] buttons.
- Drag and drop the file name.
- Use a link order specification file.

Remark You can use a link order specification file to change the display order on a file basis.
The procedures for performing this operation are described below.

(a) Generate a link order specification file

Click the [Export] button in the Link Order dialog box to open the Select Export File dialog box.

Figure 2-89. Select Export File Dialog Box



In the dialog box, specify the file (link order specification file) that is output the list of the file names displayed in [File] in the Link Order dialog box.
Click the [Save] button to generate the link order specification file.

Caution Only the file names are output to the link order specification file.
If a file with the same name exists, check the location of the file in the popup display after installing the link order specification file.

(b) Edit the link order specification file

Open the link order specification file with a editor, and then change the description order of the file names.
The code example of the link order specification file is shown below.

```
# CubeSuite+ Vx.xx.xx Link order specification file
# SampleProject: Xxxxxx xx, xxxx

file01.obj
file03.obj
library02.lib
file02.obj
library01.lib
library03.lib
    :
```

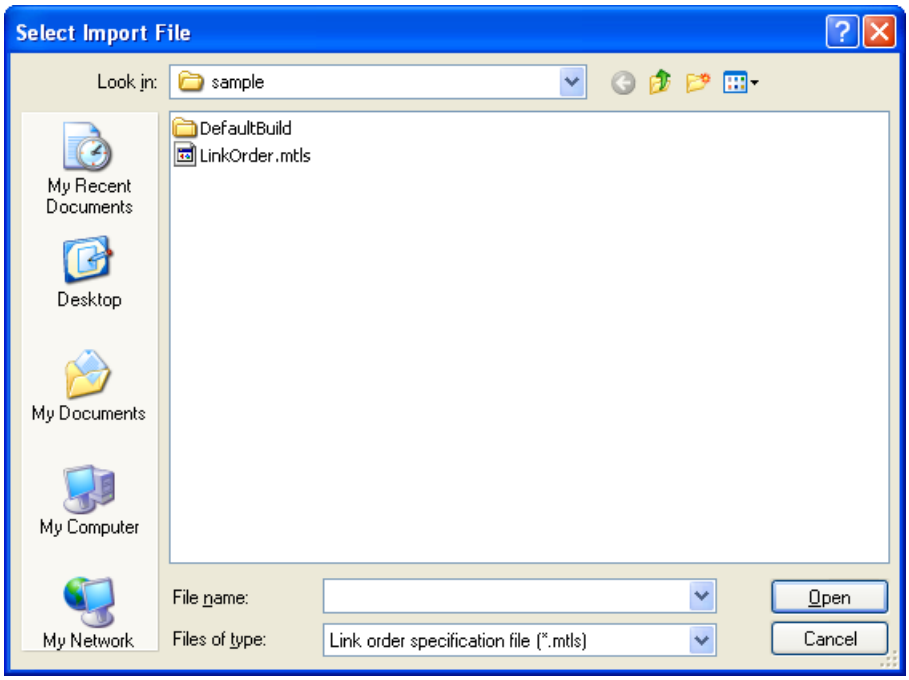
The following points should be noted when describing the link order specification file.

- Describe one file name on one line.
- Uppercase characters and lowercase characters are not distinguished for the file name.
- If the line begins with "#", the line is interpreted as a comment.
- A space or tab is ignored.

(c) Import the link order specification file

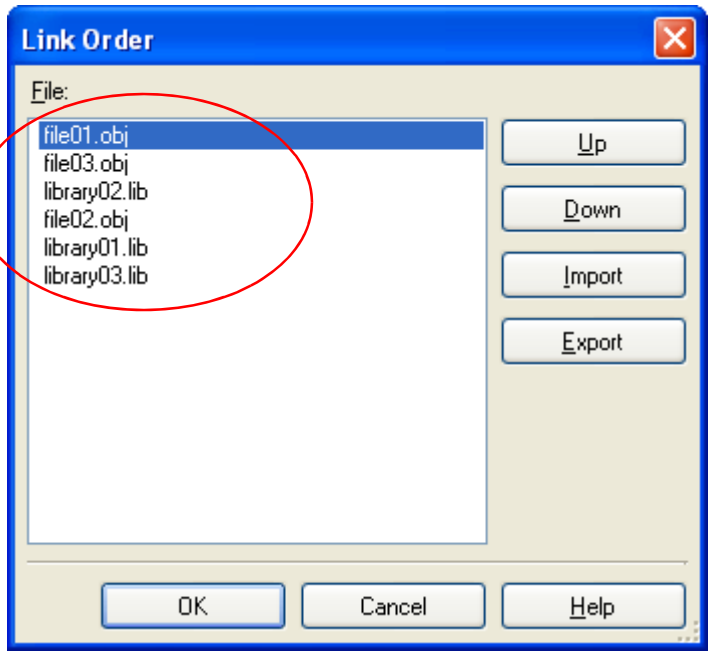
Click the [Import] button in the Link Order dialog box to open the Select Import File dialog box.

Figure 2-90. Select Import File Dialog Box



In the dialog box, select the link order specification file and click the [Open] button.
The description order of the file names are acquired from the selected link order specification file, and then they are reflected in [File] in the Link Order dialog box.

Figure 2-91. Link Order Dialog Box (After Setting Link Order)



Cautions 1. The file that is described in the link order specification file and is not added to the project is not displayed.
If the corresponding file exists, the list of the file names will be displayed in the Output panel.

2. The file that is added to the project and is not described in the link order specification file is displayed in the end of [File].

3. If a file with the same name exists, check the location of the file in the popup display (it will appear when you hover the mouse cursor over the file name).
To change the link order, use the [Up] and [Down] buttons, or drag and drop the file names.

(3) Set the file link order

Click the [OK] button in the Link Order dialog box to set the input order of the files to the linker.

2.11.3 Change the build order of subprojects

Builds are run in the order of subproject, main project, but when there are multiple subprojects added, the build order of subprojects is their display order on the project tree.

To change the display order of the subprojects on the project tree, drag the subproject to be moved and drop it on the desired location.

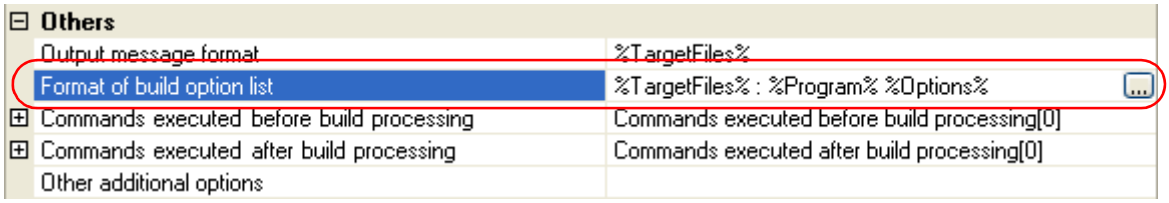
2.11.4 Display a list of build options

You can display the list of build options set currently on the Property panel for the project (main project and subproject).

Select [Build Options List] from the [Build] menu. The current settings of the options for the project will be displayed on the [Build Tool] tab from the Output panel in the build order.

Remark You can change the display format of the build option list.
Select the build tool node on the project tree and select the [Common Options] tab on the Property panel.
Set the [Format of build option list] property in the [Others] category.

Figure 2-92. [Format of build option list] Property



The following placeholders are supported.

%Program%: Replaces with the program name under execution.

%Options%: Replaces with the command line option under build execution.

%TargetFiles%: Replaces with the file name being compile/assemble or making link.

"%TargetFiles% : %Program% %Options%" is set by default.

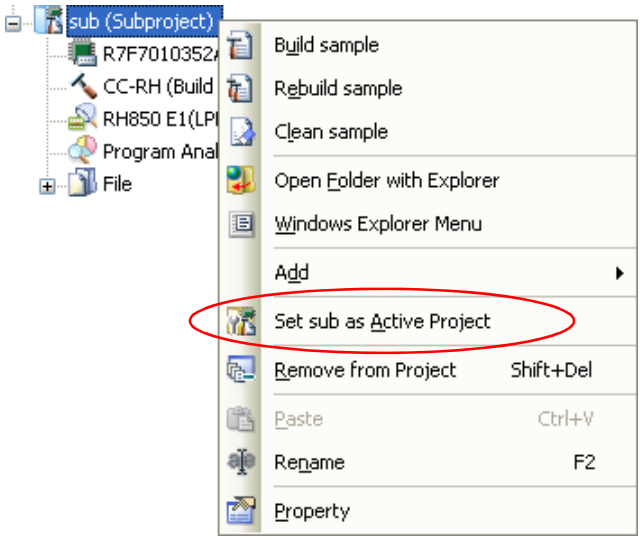
2.11.5 Change the build target project

When running a build that targets a specific project (main project or subproject), you must set that project as the "active project".

Caution **The active project will be the debug target.**
Note that the active project cannot be set during connection with the debug tool; be sure to disconnect the debug tool before setting the active project.

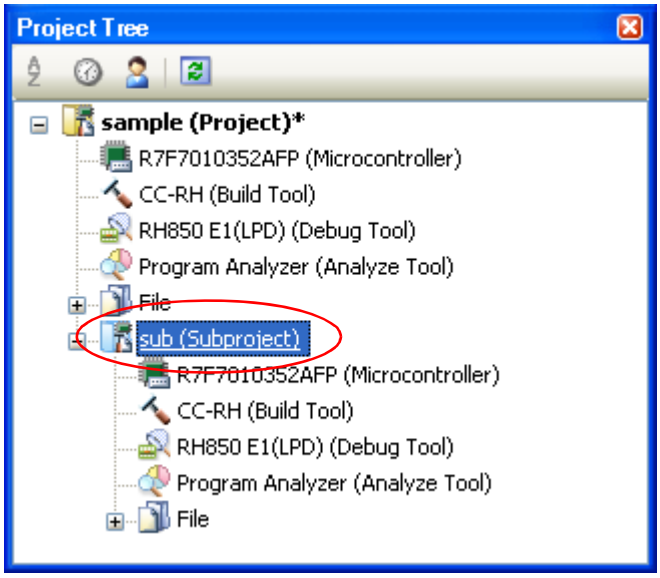
To set the active project, select the main project or subproject to be set as the active project on the project tree and select [Set *selected subproject* as Active Project] from the context menu.

Figure 2-93. [Set *selected project* as Active Project] Item



When a project is set as the active project, that project is underlined.

Figure 2-94. Active Project



Remarks 1. Immediately after creating a project, the main project is the active project.

2. When you remove a subproject that set as the active project from a project, the main project will be the active project.

Caution When a build is started by a button on the tool bar, the build is run for the entire project. To run a build only for the active project, start it from the context menu of the active project.

2.11.6 Add a build mode

When you wish to change the build options and macro definitions according to the purpose of the build, you can collectively change those settings.

Build options and macro definition settings are organized into what is called "build mode", and by changing the build mode, you eliminate the necessity of changing the build options and macro definition settings every time.

The build mode prepared by default is only "DefaultBuild". Add a build mode according to the purpose of the build.

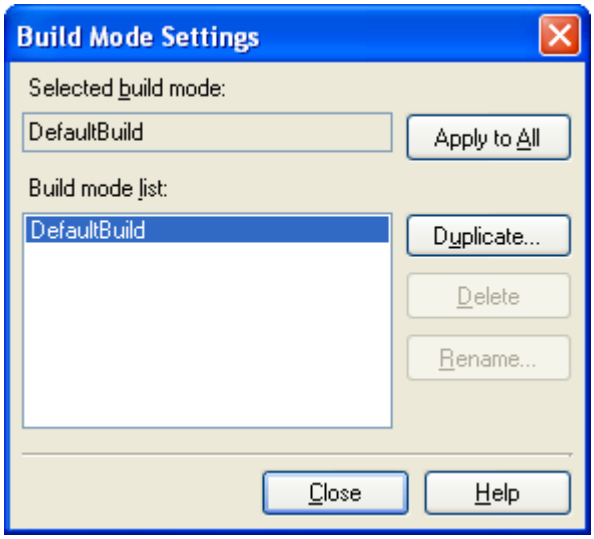
The method to add a build mode is shown below.

(1) Create a new build mode

Creating a new build mode is performed with duplicating an existing build mode.

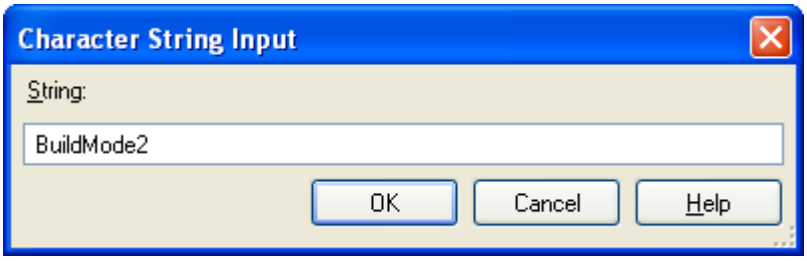
Select [Build Mode Settings...] from the [Build] menu. The Build Mode Settings dialog box will open.

Figure 2-95. Build Mode Settings Dialog Box



Select the build mode to be duplicated from the build mode list and click the [Duplicate...] button. The Character String Input dialog box will open.

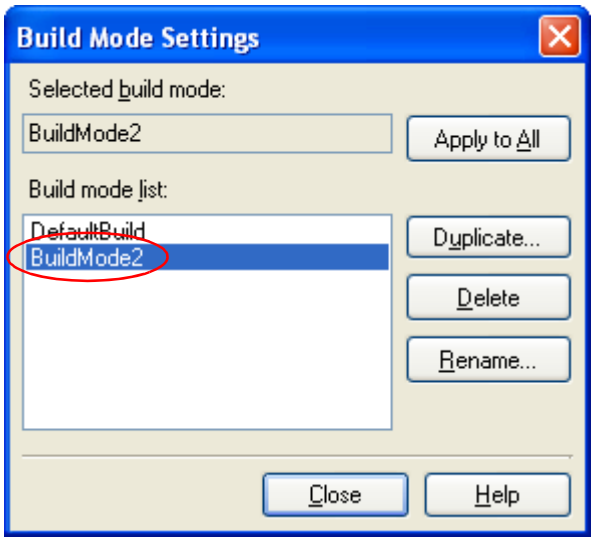
Figure 2-96. Character String Input Dialog Box



In the dialog box, enter the name of the build mode to be created and then click the [OK] button. The build mode with that name will be duplicated.

The created build mode is added to the build modes of the main project and all the subprojects which belong to the project.

Figure 2-97. Build Mode Settings Dialog Box (After Adding Build Mode)



(2) Change the build mode

Change the build mode to the newly created build mode (see "2.11.7 Change the build mode").

(3) Change the setting of the build mode

Select the build tool node on the project tree and change the build options and macro definition settings on the Property panel.

Remark Creating a build mode is regarded a project change.
When closing the project, you will be asked to confirm whether or not to save the build mode.

2.11.7 Change the build mode

When you wish to change the build options and macro definitions according to the purpose of the build, you can collectively change those settings.

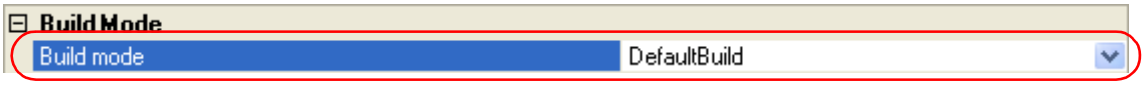
Build options and macro definition settings are organized into what is called "build mode", and by changing the build mode, you eliminate the necessity of changing the build options and macro definition settings every time.

(1) When changing the build mode for the main project or subprojects

Select the Build tool node of the target project on the project tree and select the [Common Options] tab on the Property panel.

Select the build mode to be changed to on the [Build mode] property in the [Build Mode] category.

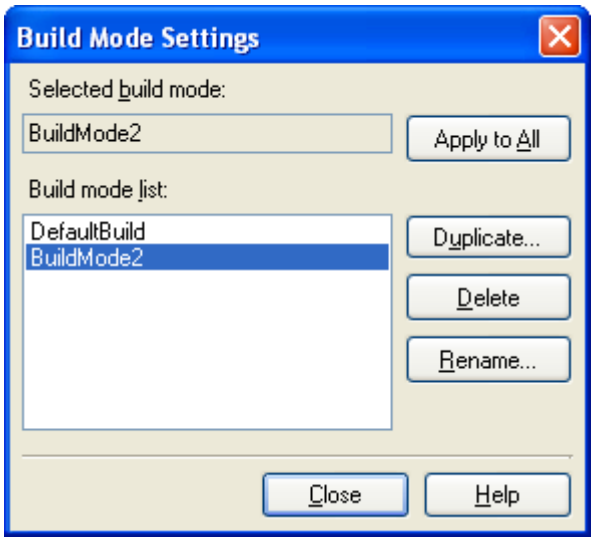
Figure 2-98. [Build mode] Property



(2) When changing the build mode for the entire project

Select [Build Mode Settings...] from the [Build] menu. The Build Mode Settings dialog box will open.

Figure 2-99. Build Mode Settings Dialog Box



Select the build mode to be changed from the build mode list. The selected build mode will be displayed in [Selected build mode].

Click the [Apply to All] button. The build mode for the main project and all the subprojects which belong to the project will be changed to the build mode selected in the dialog box.

Caution For projects that the selected build mode does not exist, the build mode is duplicated from "DefaultBuild" with the selected build mode name, and the build mode is changed to the duplicated build mode.

Remarks 1. The build mode prepared by default is only "DefaultBuild".
See "2.11.6 Add a build mode" for the method of adding a build mode.

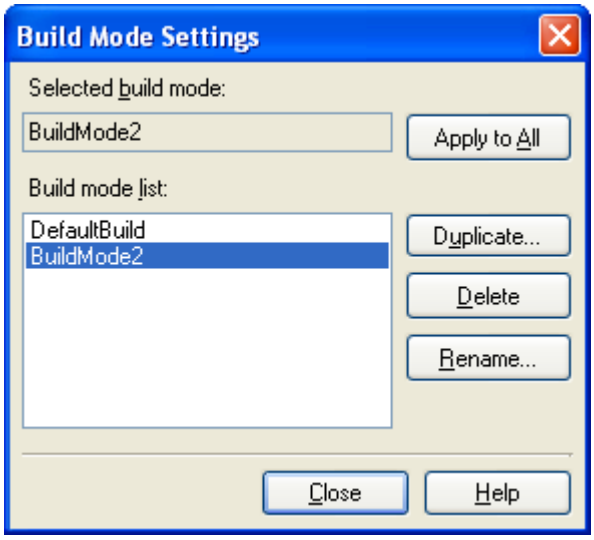
2. You can change the name of the build mode by selecting the build mode from the build mode list and clicking the [Rename...] button.
However, you cannot change the name of "DefaultBuild".

2.11.8 Delete a build mode

Deleting a build mode is performed with the Build Mode Settings dialog box.

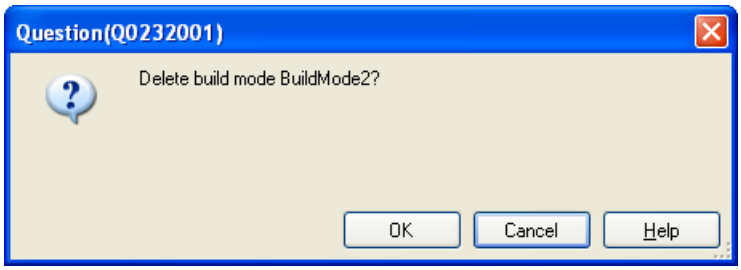
Select [Build Mode Settings...] from the [Build] menu. The dialog box will open.

Figure 2-100. Build Mode Settings Dialog Box



Select the build mode to be deleted from [Build mode list] and click the [Delete] button. The Message dialog box below will open.

Figure 2-101. Message Dialog Box



To continue with the operation, click the [OK] button in the dialog box.

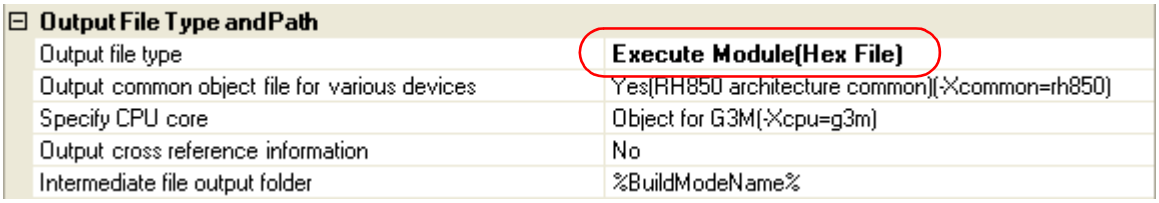
The selected build mode is deleted from the project.

Caution You cannot delete "DefaultBuild".

2.11.9 Set the current build options as the standard for the project

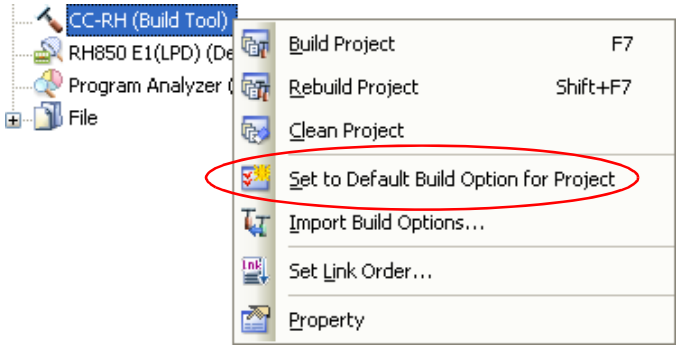
On the Property panel, if you add a change to the settings for the standard build options, the value of the property will be displayed in boldface.

Figure 2-102. Property Panel (After Changing Standard Build Option)



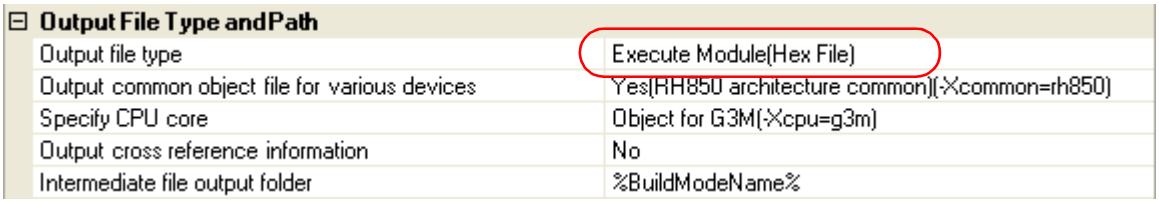
To make the build options for the currently selected project (main project or subproject) the standard build options (remove the boldface), select the Build tool node on the project tree and select [Set to Default Build Option for Project] from the context menu.

Figure 2-103. [Set to Default Build Option for Project] Item



The values of the properties after setting them as the standard build option are as shown below.

Figure 2-104. Property Panel (After Setting Standard Build Option)



Caution When the main project is selected, only the main project settings are made.Even if subprojects are added, their settings are not made.

2.12 Run a Build

This section explains operations related to running a build.

(1) Build types

The following types of builds are available.

Table 2-1. Build Types

Type	Description
Build	Out of build target files, runs a build of only updated files. -> See "2.12.1 Run a build of updated files".
Rebuild	Runs a build of all build target files. -> See "2.12.2 Run a build of all files".
Rapid build	Runs a build in parallel with the change of the build setting. -> See "2.12.3 Run a build in parallel with other operations".
Batch build	Runs builds in batch with the build modes that the project has. -> See "2.12.4 Run builds in batch with build modes".

Remarks 1. Builds are run in the order of subproject, main project.
Subprojects are built in the order that they are displayed on the project tree (see "2.11.3 Change the build order of subprojects").
Note that when the build target project depends on another project, the build of the depended project is run before the target project.

2. If there are files being edited with the Editor panel when running a build, rebuild, or batch build, then all these files are saved.

(2) Displaying execution results

The execution results of the build (output messages of the build tool) are displayed in each tab on the Output panel.

- Build, rebuild, or batch build
  -> [All Messages] tab and [Build Tool] tab
- Rapid build
  -> [Rapid Build] tab

Figure 2-105. Build Execution Results (Build, Rebuild, or Batch Build)

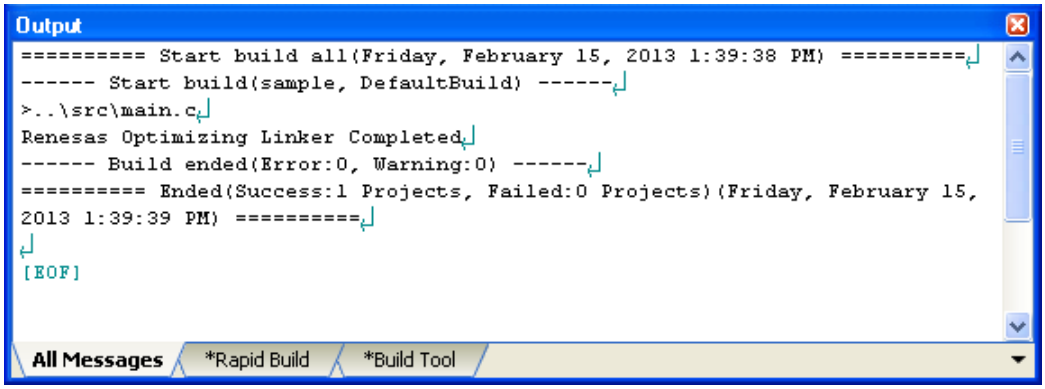
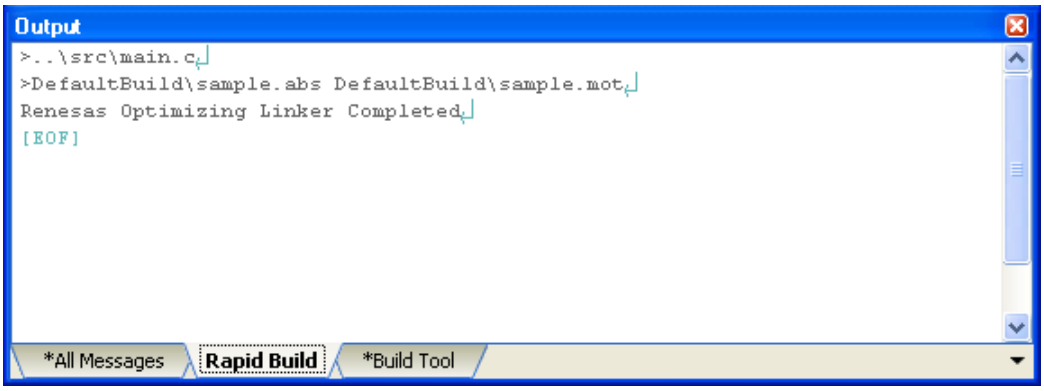


Figure 2-106. Build Execution Results (Rapid Build)



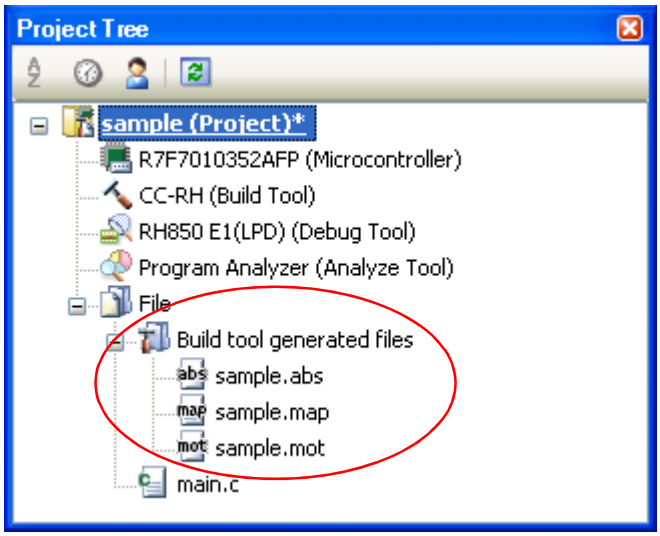
Remarks 1. The text in the [Rapid Build] tab becomes dimmed.

2. When a file name or line number can be obtained from the output messages, if you double click on the message, you can jump to the relevant line in the file.

3. If you press the [F1] key when the cursor is on a line displaying the warning or error message, you can display the help related to that line's message.

Files generated by the build tool appear under the Build tool generated files node on the Project Tree panel.

Figure 2-107. Build Tool Generated Files



Remark Files displayed under the Build tool generated files node are as follows.

- For other than the library project

Load module file (*.abs)
Intel Hex file (*.hex)
Motorola S-record file (*.mot)
Binary file (*.bin)
Link map file (*.map)
Library file (*.lib)
Library list file (*.lbp)
Error message file (*.err)

- For the library project
  Library file (*.lib)
  Error message file (*.err)

Caution The Build tool generated files node is created during build.
This node will no longer appear if you reload the project after building.

2.12.1 Run a build of updated files

Out of build target files, a build of only updated files is run (hereafter referred to as "build").
Running a build is performed for the entire project (main project and subprojects) or active project (see "2.11.5 Change the build target project").

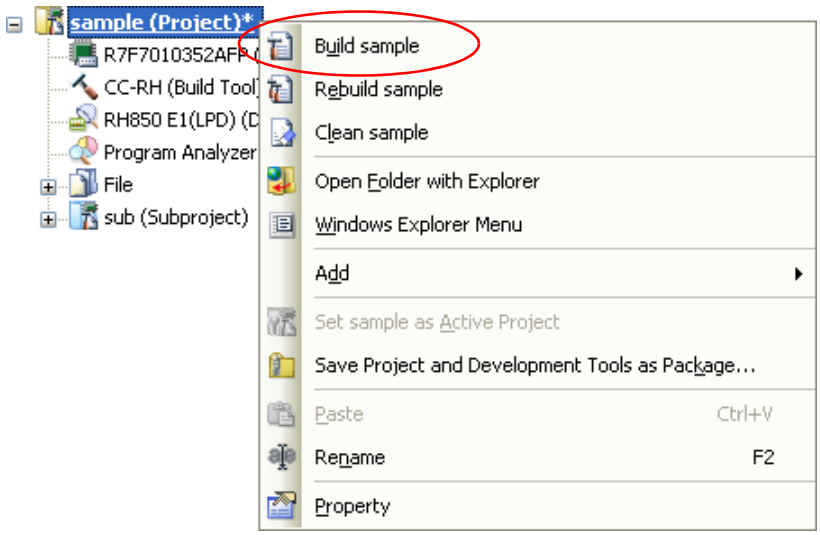
(1) When running a build of the entire project

Click on the toolbar.

(2) When running a build of the active project

Select the project, and then select [Build *active project*] from the context menu.

Figure 2-108. [Build *active project*] Item



Remark If the build of the included source file is not run after editing the header file and running the build, update the file dependencies (see "2.3.6 Update file dependencies").

2.12.2 Run a build of all files

A build of all build target files is run (hereafter referred to as "rebuild").
The cross reference file is deleted.
Running a rebuild is performed for the entire project (main project and subprojects) or active project (see "2.11.5 Change the build target project").

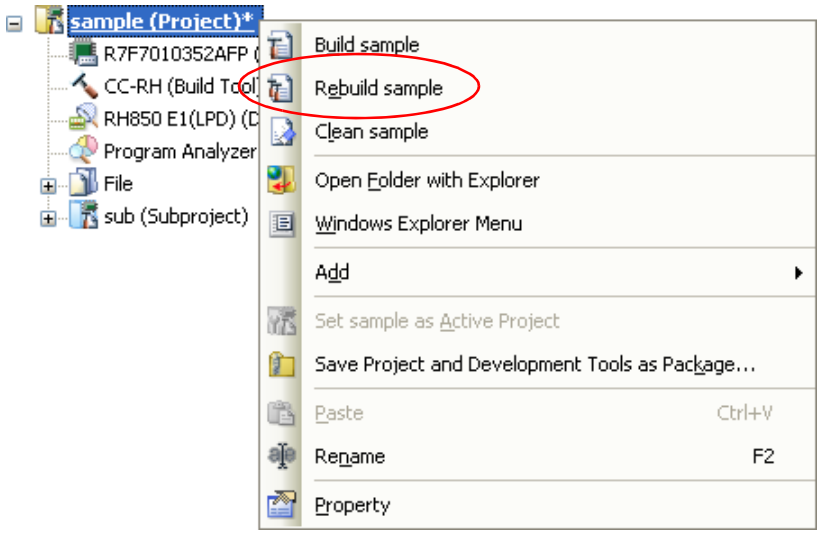
(1) When running a rebuild of the entire project

Click on the toolbar.

(2) When running a rebuild of the active project

Select the project, and then select [Rebuild *active project*] from the context menu.

Figure 2-109. [Rebuild *active project*] Item



2.12.3 Run a build in parallel with other operations

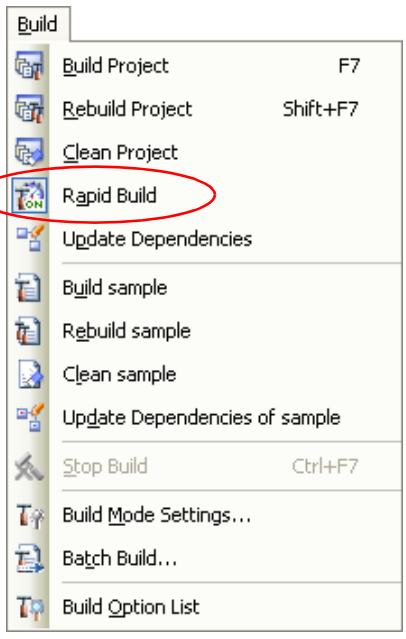
CubeSuite+ can automatically start a build when one of the following events occurs (hereafter referred to as "rapid build").

- When any one of the following files that are added to the project is updated: (C source file, assembly source file, symbol address file, header file, object file, relocatable file, and library file)
- When a build target file has been added to or removed from the project
- When the link order of object module files and library files is changed
- When the property of the build tool or build target file is changed

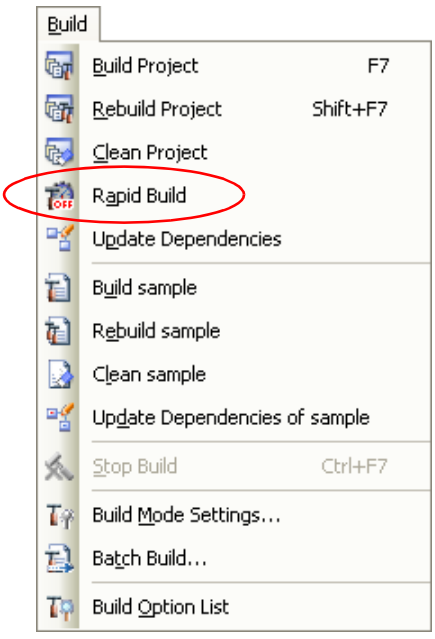
If a rapid build is enabled, it is possible to perform a build in parallel with the above operations.
To enable/disable a rapid build, select [Rapid Build] from the [Build] menu.A rapid build is enabled by default.

Figure 2-110. [Rapid Build] Item

[When a rapid build is valid]



[When a rapid build is invalid]



Remarks 1. After editing source files, it is recommend to save frequently by pressing the [Ctrl] + [S] key.

2. Enable/Disable setting of the rapid build applies to the entire project (main project and subprojects).

3. If you disable a rapid build while it is running, it will be stopped at that time.

Caution This function will also be enabled in an external text editor if the [Observe registered files changing] checkbox is selected in the [Build/Debug] category of the Option dialog box.

2.12.4 Run builds in batch with build modes

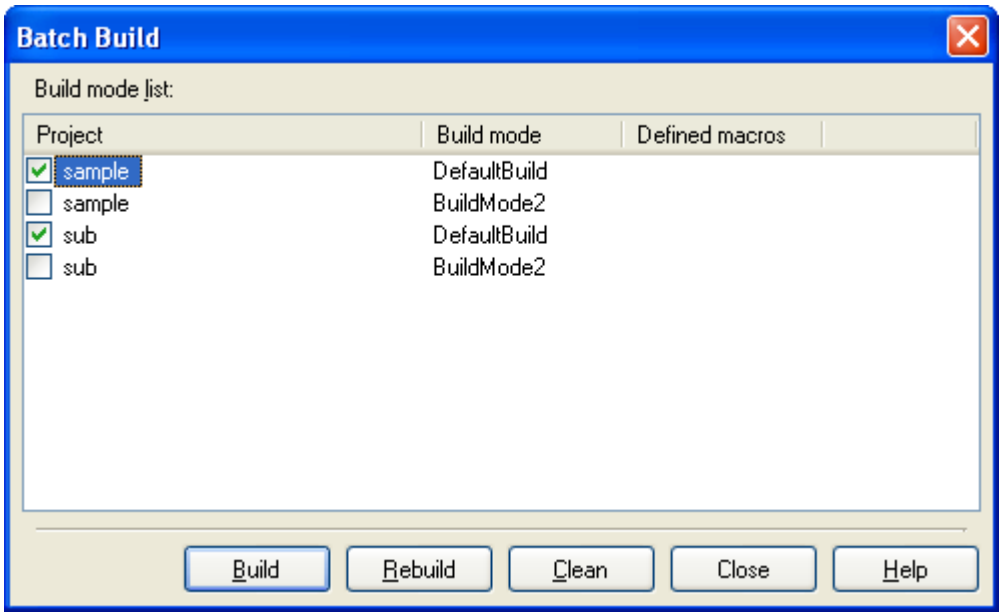
You can run builds, rebuilds and cleans in batch with the build modes that the project (main project and subproject) has (hereafter referred to as "batch build").

Remark See the sections below for a build, rebuild, and clean.

- Build: See "2.12.1 Run a build of updated files".
- Rebuild: See "2.12.2 Run a build of all files".
- Clean: See "2.12.9 Delete intermediate files and generated files".

Select [Batch Build] from the [Build] menu. The Batch Build dialog box will open.

Figure 2-111. Batch Build Dialog Box



In the dialog box, the list of the combinations of the names of the main project and subprojects in the currently opened project and their build modes and macro definitions is displayed.

Select the check boxes for the combinations of the main project and subprojects and build modes that you wish to run a batch build, and then click the [Build], [Rebuild], or [Clean] button.

Remark The batch build order follows the project build order, the order of the subprojects, main project.
When multiple build modes are selected for a single main project or subproject, after running builds of the subproject with all the selected build modes, the build of the next subproject or main project is run.

2.12.5 Compile/assemble/link multiple files simultaneously

If there are multiple build target files, you can compile, assemble, and link all files simultaneously with a single ccrh command call (hereafter referred to as "simultaneous build").

An image of calling the ccrh command is shown below.

Example When build target files are aaa.c, bbb.c, and ccc.c

- When a build is run simultaneously

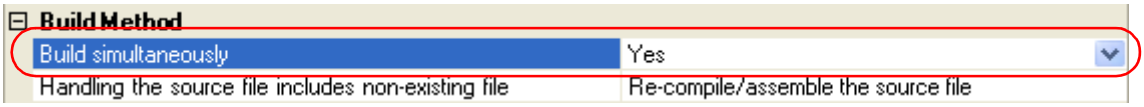
```
>ccrh -Xcommon=v850e3v5 aaa.c bbb.c ccc.c <- "aaa.abs" is generated.
```

- When a build is not run simultaneously

```
>ccrh -Xcommon=v850e3v5 aaa.c <- "aaa.obj" is generated.
>ccrh -Xcommon=v850e3v5 bbb.c <- "bbb.obj" is generated.
>ccrh -Xcommon=v850e3v5 ccc.c <- "ccc.obj" is generated.
>ccrh -Xcommon=v850e3v5 aaa.obj bbb.obj ccc.obj <- "aaa.abs" is generated.
```

Whether to run a build simultaneously is made with the property.
Select the build tool node on the project tree and select the [Common Options] tab on the Property panel.
Select [Yes] in the [Build simultaneously] property in the [Build Method] category.

Figure 2-112. [Build simultaneously] Property



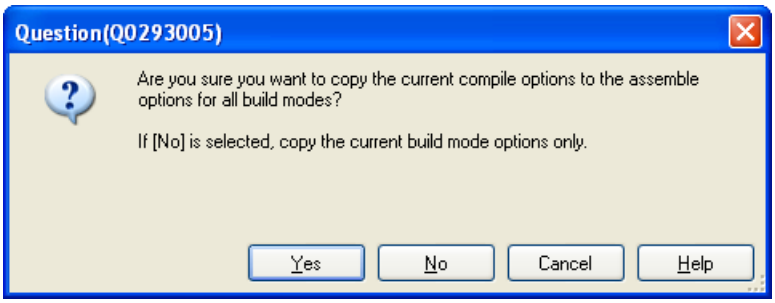
At this time, the [Assemble Options] tab is hidden.
The assembly source file (except the file with the individual options) is assembled using the settings of the [Compile Options] tab.

Remarks 1. The build of the files with the individual build options and files to be executed prior to the build is run simultaneously.
A build of the file that is not targeted for a simultaneous build is run separately.

2. If the source file is older than the generated object module file or related properties and project or the like, the object module file will be used for the build instead of the source file.

3. If the [Build simultaneously] property is changed from [Yes] to [No], the following message dialog box will open.

Figure 2-113. Message Dialog Box



When [Yes] is clicked in the dialog box, the same [Compile Options] tab settings are copied to the [Assemble Options] tab in all build modes.
When [No] is clicked, the same [Compile Options] tab settings are copied to the [Assemble Options] tab in the current build mode.
If [Cancel] is clicked, the state before the [Assemble Options] tab was hidden will be displayed.

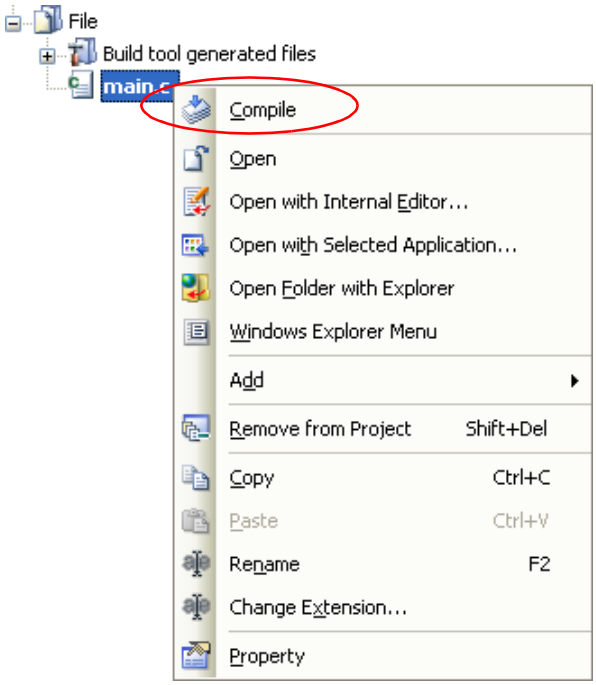
2.12.6 Compile/assemble individual files

You can just compile or assemble for each source file added to the project.

(1) When compiling a C source file

Select the C source file on the project tree and select the [Compile] from the context menu.

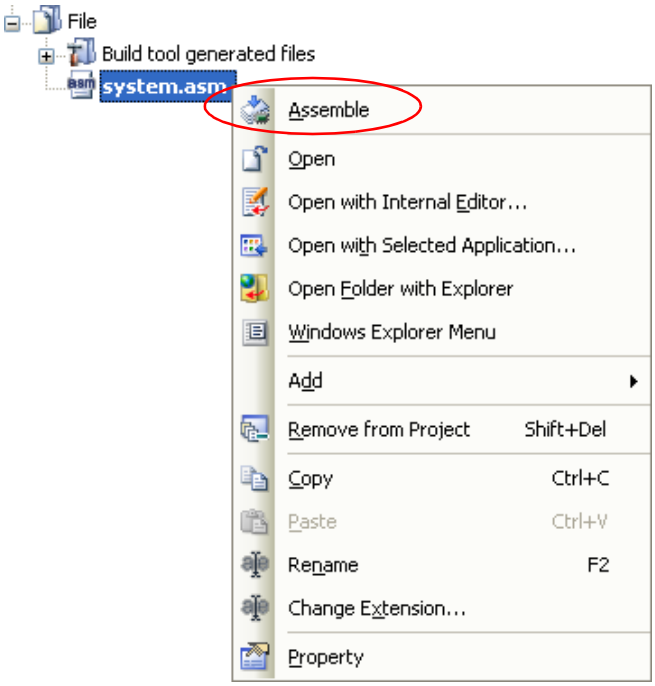
Figure 2-114. [Compile] Item



(2) When assembling an assembly source file

Select the assembly source file on the project tree and select the [Assemble] from the context menu.

Figure 2-115. [Assemble] Item



2.12.7 Stop running a build

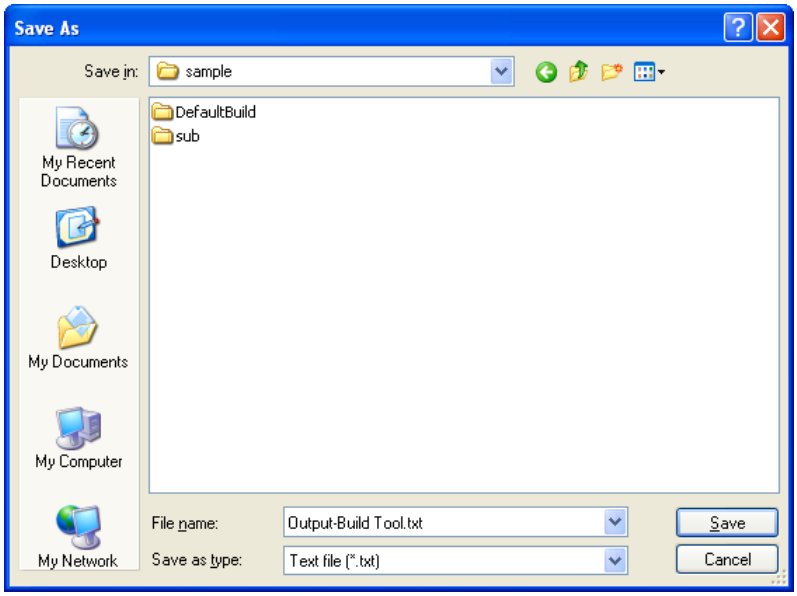
To stop running a build, rebuild, or batch build, click on the toolbar.

2.12.8 Save the build results to a file

You can save the execution results of the build (output messages of the build tool) that displayed on the Output panel.

Select the [Build Tool] tab on the panel, and then select [Save Output - Build Tool As...] from the [File] menu. The Save As dialog box will open.

Figure 2-116. Save As Dialog Box



In the dialog box, specify the file to be saved and then click the [Save] button.

2.12.9 Delete intermediate files and generated files

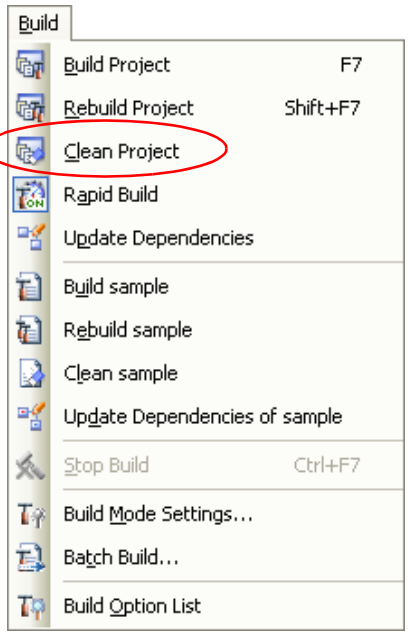
You can delete all the intermediate files and generated files output by running a build (hereafter referred to as "clean").

Running a clean is performed for the entire project (main project and subprojects) or active project (see "2.11.5 Change the build target project").

(1) When running a clean of the entire project

From the [Build] menu, select [Clean Project].

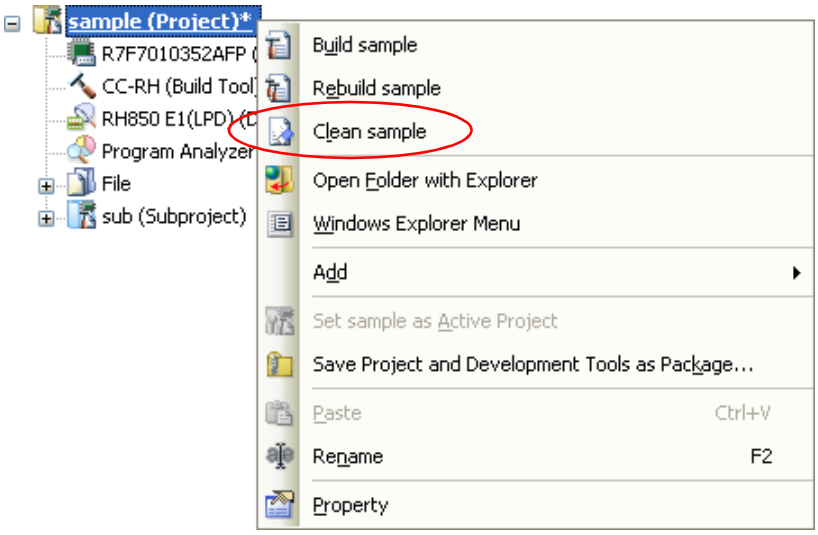
Figure 2-117. [Clean Project] Item



(2) When running a clean of the active project

Select the project, and then select [Clean *active project*] from the context menu.

Figure 2-118. [Clean *active project*] Item



2.13 Estimate the Stack Capacity

To estimate the stack capacity, use Call Walker.

Call Walker performs a static analysis, and displays the symbols and their callers in a tree format, as well as stack information for each symbol (symbol name, attribute, address, size, stack size, and file name) in list format.

To start Call Walker, select Windows [Start] >> [All programs] >> [Renesas Electronics CubeSuite+] >> [Utilities] >> [Call Walker].

Remark In Windows 8, double-click on [Call Walker] on the start screen.

To exit from Call Walker, select Call Walker [File] menu >> [Exit].

See Call Walker [Help] menu >> [Help Topics] for Call Walker operations.

CHAPTER 3 BUILD OUTPUT LISTS

This chapter explains the format and other aspects of lists output by a build via each command.

3.1 Assemble List File

This section explains the assemble list file.

The assemble list is the list-formatted version of the code that is output when the source has been compiled and assembled.

It can be used to check the code resulting from compilation and assembly.

The methods for outputting the assemble list file are shown below.

(1) For CubeSuite+

Select the build tool node on the Project Tree panel and select the [Compile Options] tab on the Property panel.
To output the assemble list file, select [Yes(-Xasm_option=-Xprn_path)] in the [Output assemble list file] property in the [Assemble List] category.
Specify the output destination in the [Output folder for assemble list file] property.
The file name will be the source file name with the extension replaced by ".prn".

When [Yes(-Xpass_source)] in the [Output comment to assembly source file] property in the [Output Code] category is selected, the C source program that corresponds to the assembly source program is output as comments in the assemble list file.

(2) Command line

When the -Xprn_path option is specified, the assemble list file is output under the source file name with the extension replaced by ".prn".
Also, the output folder and file name can be specified by the parameter of the -Xprn_path option.

If the -Xpass_source option is specified at the same time, the C source program that corresponds to the assembly source program is output as comments in the assemble list file.

3.1.1 Structure of the assemble list

The structure and contents of the assemble list are shown below.

Output Information	Description
Assemble list	Location counter value, code, line number, and source program under assembly
Section list	Type, size, and name of section
Command line information	Character string of command line of assembler

3.1.2 Assemble list

The location counter value, code, line number, and source program under assembly is output.

The output example of the assemble list is shown below.

```
(1)       (2)                 (3) (4)
OFFSET    CODE                NO  SOURCE STATEMENT


00000000                       1  #CC-RH Compiler RH850 Assembler Source
00000000                       2  #@      CC-RH Version : VX.XX.XXx [DD Mmm YYY]
00000000                       3  #@      Command : main.c -Xcommon=v850e3v5 -S
00000000                       4  #@      compiled at Sun Jan 1 00:00:00 2012
00000000                       5          .cseg text
00000000                       6          ld.w $_data,r12
00000000  440E0000             -- movhi       0x0,gp,r1
00000004  21670100             -- ld.w        0x0[r1],r12
00000008                       7
00000000                       8          .dseg data
00000000  00000000             9  _data:  .dw 0
00000004                      10
```

Number	Description
(1)	Location counter value The location counter value for the beginning of the code generated for the source program of the corresponding line is output.
(2)	Code The code (machine language instruction or data) generated for the source program of the corresponding line is output. Each byte is expressed as 2-digit hexadecimal number.
(3)	Line number The number of the line is output. This is expressed in a decimal number.
(4)	Source program The source program of the line is output. If instruction expansion is performed for the instruction at that line, the disassembly of the array of machine language instructions generated after the instruction expansion is displayed after "--". Compiler information (lines 1 to 4) is output only when an assembly source file output from the compiler is assembled.

3.1.3 Section list

The type, size, and name of the section is output.

The output example of the section list is shown below.

```
Section List
(1)     (2)                 (3)
Attr    Size                Name

TEXT            8 (00000008) .text
DATA            4 (00000004) .data
```

Number	Description
(1)	Section type The type of the section is output as the relocation attribute.
(2)	Section size The size of the section is output. This is expressed in a decimal number and also expressed in hexadecimal number in parentheses.
(3)	Section name The name of the section is output.

3.1.4 Command line information

The character string of the command line of the assembler is output.

The output example of the command line information is shown below.

```
Command Line Parameter
a.asm -Xcommon=v850e3v5 -Xprn_path      (1)
```

Number	Description
(1)	Character string of command line The character string of the command line specified for the assembler is output.

3.2 Link Map File

This section explains the link map file.

The link map has information of the link result. It can be referenced for information such as the section's allocation addresses.

The methods for outputting the link map file are shown below.

(1) For CubeSuite+

Select the build tool node on the Project Tree panel and select the [Link Options] tab on the Property panel.

To output the link map file, select [Yes] in the [Output link map file] property in the [List] category.

Specify the output destination in the [Output folder] property in the [Output File] category.

It is also shown on the project tree, under the Build tool generated files node.

~~The file name will be the output file name with the extension replaced by ".map".~~

The file name will be the project file name with the extension replaced by ".map".

(2) Command line

When the -form=library or -extract option is not specified and the -list option is specified, the link map file is output to the current folder under the output file name with the extension replaced by ".map".

The output information for the link map file can be specified by specifying the -show option at the same time.

See "3.2.1 Structure of link map" for details.

3.2.1 Structure of link map

The structure and contents of the link map are shown below.

Output Information	Description	-show Option Specification	When -show Option Is Omitted
Option information	Option strings specified by a command line or subcommand file	-	Output
Error information	Error message	-	Output
Link map information	Section name, start/end addresses, size, and type	-	Output
Total section size	Total sizes of RAM, ROM, and program sections	-show=total_size	No output
Symbol information	Static defined symbol name, address, size, type (in the order of address), and whether optimization is applied When the -show=reference is specified, the reference count of each symbol is also output.	-show=symbol -show=reference	No output
Cross reference information	Symbol reference information	-show=xreference	No output

Caution The -show option is valid when the -list option is specified.
See "-SHow" for details about the -show option.

3.2.2 Option information

Option strings specified by a command line or subcommand file are output.

The output example of the option information when the following command line and subcommand file are specified is shown below.

<Command line>

```
>rlink -subcommand=test.sub -list -show
```

<Subcommand file "test.sub">

```
input sample.obj
```

```
*** Options ***

-subcommand=test.sub     (1)
input sample.obj         (2)
-list                    (1)
-show                    (1)
```

Number	Description
(1)	Options specified by command line The options specified by the command line are output (in their specified order).
(2)	Options specified in subcommand file The options specified in subcommand file "test.sub" are output.

3.2.3 Error information

Error messages are output.

The output example of the error information is shown below.

```
*** Error Information ***

** E0562310 Undefined external symbol "_func_02" referenced in "sample.obj"      (1)
```

Number	Description
(1)	Error message Error messages are output.

3.2.4 Link map information

Start/end addresses, size, and type of each section are output in the order of address.
The output example of the link map information is shown below.

```
*** Mapping List ***

(1)                                (2)        (3)         (4)      (5)
SECTION                            START      END         SIZE     ALIGN

.text
                                   00000000   0000003b      3c     2
.data
                                   fe600006   fe600003       4     4
.bss
                                   fe600004   fe60000b       8     4
```

Number	Description
(1)	Section name The name of the section is output.
(2)	Start address The start address is output. This is expressed in a hexadecimal number.
(3)	End address The end address is output. This is expressed in a hexadecimal number.
(4)	Section size The section size is output (byte). This is expressed in a hexadecimal number.
(5)	Section alignment size The section alignment size is output.

3.2.5 Total section size

When the -show=total_size option is specified, the total sizes of RAM, ROM, and program sections are output.
The output example of the total section size is shown below.

```
*** Total Section Size ***

RAMDATA SECTION:         00000660 Byte(s) (1)
ROMDATA SECTION:         00000174 Byte(s) (2)
PROGRAM SECTION:         000016d6 Byte(s) (3)
```

Number	Description
(1)	Total size of RAM data sections The total size of RAM data sections is output. This is expressed in a hexadecimal number.

Number	Description
(2)	Total size of ROM data sections The total size of ROM data sections is output. This is expressed in a hexadecimal number.
(3)	Total size of program sections The total size of program sections is output. This is expressed in a hexadecimal number.

3.2.6 Symbol information

When the -show=symbol option is specified, the external defined symbol or static internal defined symbol address, size, type, and whether optimization is applied are output in the order of address.

When the -show=reference option is specified, the reference count of each symbol is also output.

The output example of the symbol information is shown below.

```
*** Symbol List ***

SECTION=(1)
FILE=(2)
                (3)          (4)              (5)
                START        END              SIZE
  (6)           (7)          (8)              (9)           (10)    (11)
  SYMBOL        ADDR         SIZE             INFO          COUNTS  OPT

SECTION=.text
FILE=sample.obj
                00000000     00000023         24
  _main
                00000000            0         func ,g            0
  _func_01
                00000018            0         func ,g            0
SECTION=.bss
FILE=sample.obj
                fe600004     fe60000b         8
  _gvall
                fe600004            4         data ,g            0
```

Number	Description
(1)	Section name The name of the section is output.
(2)	File name The file name is output.
(3)	Start address The start address of the corresponding section included in the file shown in (2) is output. This is expressed in a hexadecimal number.

Number	Description
(4)	End address The end address of the corresponding section included in the file shown in (2) is output. This is expressed in a hexadecimal number.
(5)	Section size The size of the corresponding section included in the file shown in (2) is output (in byte units). This is expressed in a hexadecimal number.
(6)	Symbol name The symbol name is output.
(7)	Symbol address The symbol address is output. This is expressed in a hexadecimal number.
(8)	Symbol size The symbol size is output (in byte units). This is expressed in a hexadecimal number.
(9)	Symbol type The data type and declaration type are output. - Data type func: Function name data: Variable name entry: Entry function name none: Undefined (label, assembler symbol) - Declaration type g: External definition l: Internal definition
(10)	Reference count of symbol The reference count of the symbol is output. This is expressed in a hexadecimal number. This item is output only when the -show=reference option is specified. When the reference count of the symbol is not output, "*" is output.
(11)	Whether optimization is applied Whether optimization is applied is output. ch: Symbol changed by optimization cr: Symbol generated by optimization mv: Symbol moved by optimization

3.2.7 Cross reference information

When the -show=xreference option is specified, the reference information of symbols (cross reference information) is output.

The output example of the cross reference information is shown below.

```
*** Cross Reference List ***

(1)  (2)         (3)             (4)        (5)
No   Unit Name   Global.Symbol   Location   External Information
0001 sample1
     SECTION=.text
                 _main
                                 00000000
                 _func_01
                                 00000018
     SECTION=.data
                 _gval3
                                 fe600000   0003(00000032:.text)
                                            0003(00000038:.text)
     SECTION=.bss
                 _gval1
                                 fe600004   0001(0000001a:.text)
                                            0001(00000020:.text)
                 _gval2
                                 fe600008   0002(00000026:.text)
                                            0002(0000002c:.text)
0002 sample2
     SECTION=.text
                 _func02
                                 00000024   0001(0000000a:.text)
0003 sample3
     SECTION=.text
                 _func03
                                 00000030   0001(00000010:.text)
```

Number	Description
(1)	Unit number The identification number in object units is output.
(2)	Object name The object name is output in the order of input when linking.
(3)	Symbol name The symbol name is output in the ascending order of allocation address for each section.
(4)	Symbol allocation address The symbol allocation address is output. When the -form=relocate option is specified, this is a relative value from the start of the section.
(5)	Address of external symbol that has been referenced The address of the external symbol that has been referenced is output. *Unit number* (*address or offset in section*:*section name*)

3.3 Link Map File (When Objects Are Combined)

This section explains the link map file that is output when the object combine function (see "2.8.3 Combine multiple objects") is used in a multi-core project.

The link map file has detailed information regarding object combining.

The methods for outputting the link map file are shown below.

(1) For CubeSuite+

The settings for outputting the link map file should be made in the boot loader project.

Select the build tool node on the Project Tree panel and select the [Link Options] tab on the Property panel.

To output the link map file, select [Yes] in the [Output link map file] property in the [List] category.

The file is output to the folder specified in the [Output folder for united hex file] property in the [Output File] category from the [Hex Output Options] tab.

The output file name will be the file name specified in the [Output file name] property with the extension replaced by ".map".

(2) Command line

Specify one or more Intel HEX files or Motorola S-record files as the input files.

Caution The input files should have the same type (an Intel HEX file and a Motorola S-record file cannot be combined together).

When the -list option is specified, the link map file is output to the current folder.

At this time, specify the -form=hexadecimal option when Intel HEX files are specified, or the -form=stype option when Motorola S-record files are specified.

The output file name is "*first input file name*_combine.map".

3.3.1 Structure of link map

The structure and contents of the link map are shown below.

Output Information	Description
Header information	Version information of the optimizing linker and time of linkage
Option information	Option strings specified by a command line or subcommand file
Error information	Error message
Entry information	Execution start address
Combined address information	Combined source files, and start and end addresses and size of continuous range data
Address overlap information	Overlapped combine source files, and start and end addresses and size of overlapped range data

3.3.2 Header information

The version information and the time of linkage are output.

The output example of the header information is shown below.

```
Renesas Optimizing Linker (VX.XX.XX)            XX-Xxx-XXXX XX:XX:XX    (1)
```

Number	Description
(1)	Version information of the optimizing linker and time of linkage The version information of the optimizing linker and the time of linkage are output.

3.3.3 Option information

Option strings specified by a command line or subcommand file are output.

The output example of the option information when the following command line and subcommand file are specified is shown below.

<Command line>

```
>rlink -subcommand=test.sub -list
```

<Subcommand file "test.sub">

```
input sample1.mot
input sample2.mot
form stype
output result
```

```
*** Options ***

-subcommand=test.sub    (1)
input sample1.mot       (2)
input sample2.mot       (2)
form stype              (2)
output result           (2)
-list                   (1)
```

Number	Description
(1)	Options specified by command line The options specified by the command line are output (in their specified order).
(2)	Options specified in subcommand file The options specified in subcommand file "test.sub" are output.

3.3.4 Error information

Error messages are output.

The output example of the error information is shown below.

```
*** Error Information ***

E0562420:"sample1.mot" overlap address "sample2.mot" : "00000100"       (1)
```

Number	Description
(1)	Error message Error messages are output.

3.3.5 Entry information

The execution start address is output.

The output example of the entry information is shown below.

```
*** Entry address ***
00000100        (1)
```

Number	Description
(1)	Execution start address The execution start address is output. However, if the execution start address is "00000000", it is not output.

3.3.6 Combined address information

The combined source files, and the start and end addresses and size of the continuous range data are output.

The output example of the combined address information is shown below.

```
*** Combine information ***
(1)             (2)         (3)         (4)
FILE            START       END         SIZE
sample1.mot
                00000100    00000127    28
sample1.mot
                00000200    00000227    28
sample2.mot
                00000250    00000263    14
sample2.mot
                00000300    0000033b    3c
```

Number	Description
(1)	Combined source file name The combined source file name is output.

Number	Description
(2)	Start addresses of continuous range data The start addresses of the continuous range data are output. This is expressed in a hexadecimal number.
(3)	End addresses of continuous range data The end addresses of the continuous range data are output. This is expressed in a hexadecimal number.
(4)	Size of continuous range data The size of the continuous range data is output (in byte units). This is expressed in a hexadecimal number.

3.3.7 Address overlap information

The overlapped combine source files, and the start and end addresses and size of the continuous range data are output.

The output example of the address overlap information is shown below.

```
*** Conflict information ***
(1)             (2)         (3)         (4)
FILE            START       END         SIZE
Conflict 1
                00000200    00000213    14
sample1.mot
sample2.mot
```

Number	Description
(1)	Overlapped combine source file name The overlapped combine source file name is output.
(2)	Start addresses of overlapped range data The start addresses of the overlapped range data are output. This is expressed in a hexadecimal number.
(3)	End addresses of overlapped range data The end addresses of the overlapped range data are output. This is expressed in a hexadecimal number.
(4)	Size of overlapped range data The size of the overlapped range data is output (in byte units). This is expressed in a hexadecimal number.

3.4 Library List File

This section explains the library list file.
The library list has information from the library creation result.

The methods for outputting the library list file are shown below.

(1) For CubeSuite+

Select the build tool node on the Project Tree panel and select the [Create Library Options] tab on the Property panel.
To output the library list file, select [Yes] in the [Output link map file] property in the [List] category.
Specify the output destination in the [Output folder] property in the [Output File] category.
It is also shown on the project tree, under the Build tool generated files node.
~~The file name will be the output file name with the extension replaced by ".lbp".~~
The file name will be the project file name with the extension replaced by ".lbp".

(2) Command line

When the -form=library or -extract option is specified and the -list option is specified, the library list file is output to the current folder under the output file name with the extension replaced by ".lbp".
The output information for the library list file can be specified by specifying the -show option at the same time.
See "3.4.1 Structure of the library list" for details.

3.4.1 Structure of the library list

The structure and contents of the library list are shown below.

Output Information	Description	-show Option Specification	When -show Option Is Omitted
Option information	Option strings specified by a command line or subcommand file	-	Output
Error information	Error message	-	Output
Library information	Library information	-	Output
Module, section, and symbol information within the library	Module within the library	-	Output
	Symbol names within a module	-show=symbol	No output
	Section names and symbol names within each module	-show=section	No output

Caution The -show option is valid when the -list option is specified.
See "-SHow" for details about the -show option.

3.4.2 Option information

Option strings specified by a command line or subcommand file are output.
The output example of the option information when they are specified by a command line or subcommand file as follows is shown below.

<Command line>

```
>rlink -subcommand=test.sub -list -show
```

<Subcommand file "test.sub">

```
form library
input extmod1
input extmod2
output usrlib.lib
```

```
*** Options ***

-subcommand=test.sub     (1)
form library             (2)
input extmod1            (2)
input extmod2            (2)
output usrlib.lib        (2)
-list                    (1)
-show                    (1)
```

Number	Description
(1)	Options specified by command line The options specified by the command line are output (in their specified order).
(2)	Options specified in subcommand file The options specified in subcommand file "test.sub" are output.

3.4.3 Error information

Messages for errors or warnings are output.

The output example of the error information is shown below.

```
*** Error Information ***

** E0561200 Backed up file "sample1.lib" into "usrlib.lbk"      (1)
```

Number	Description
(1)	Message The message is output.

3.4.4 Library information

The type of the library is output.

The output example of the library information is shown below.

```
*** Library Information ***

LIBRARY NAME=usrlib.lib (1)
CPU=RH850               (2)
ENDIAN=Little           (3)
ATTRIBUTE=user          (4)
NUMBER OF MODULE=2      (5)
```

Number	Description
(1)	Library name The library name is output.
(2)	Microcontroller name The microcontroller name is output.
(3)	Endian type The endian type is output.
(4)	Library file attribute Either a system library or user library is output.
(5)	Number of modules within library The number of modules within the library is output.

3.4.5 Module, section, and symbol information within the library

Modules within the library is output.
When the -show=symbol option is specified, symbol names within the module is output.
When the -show=section option is specified, section names within the module is also output.
The output example of the module, section, and symbol information within the library is shown below.

```
*** Library List ***

(1)             (2)
MODULE          LAST UPDATE
  (3)
  SECTION
    (4)
    SYMBOL
extmod1
                12-Dec-2011 16:30:00
  .text
    _func_01
    _func_02
extmod2
                12-Dec-2011 16:30:10
  .text
    _func_03
    _func_04
```

Number	Description
(1)	Module name The module name is output.
(2)	Module definition date The module definition date is output. If the module is updated, the date of the latest update is output.
(3)	Name of section within module The name of the section within the module is output.
(4)	Name of symbol within section The name of the symbol within the section is output.

3.5 Intel HEX File

This section explains the Intel HEX file.

The methods for outputting the Intel HEX file are shown below.

(1) For CubeSuite+

Select the build tool node on the Project Tree panel and select the [Hex Output Options] tab on the Property panel.
To output the Intel Hex file, select [Intel HEX file(-FOrm=Hexadecimal)] in the [Hex file format] property in the [Hex Format] category.
Specify the output destination in the [Output folder] property and [Output file name] property in the [Output File] category.
It is also shown on the project tree, under the Build tool generated files node.

(2) Command line

When the -form=hexadecimal option is specified, the Intel HEX file is output to the current folder under the output file name with the extension replaced by ".hex".

3.5.1 Structure of the Intel HEX file

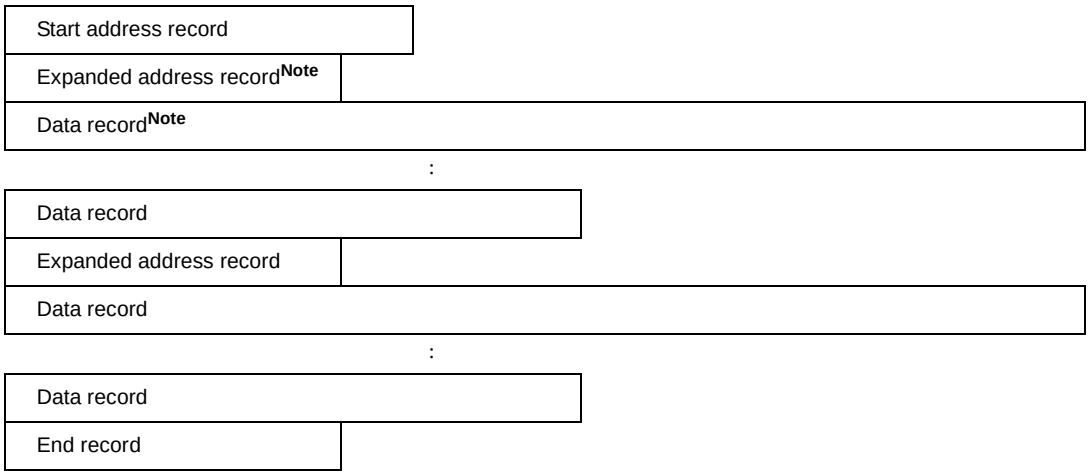
The Intel HEX file (20 bits) consists of four records[Note]: start address record, expanded address record, data record, and end record.

The Intel HEX file (32 bits) consists of six records[Note]: start linear address record, expanded linear start address record, start address record, expanded address record, data record, and end record.

Note Each record is output in ASCII code.

The structure and contents of the Intel HEX file are shown below.

Figure 3-1. Structure of Intel HEX File



Note The expanded address record and data record are repeated.

Output Information	Description
Start linear address record	Linear address
Expanded linear address record	Upper 16-bit address at bits 32 to 16

Output Information	Description
Start address record	Entry point address
Expanded address record	Paragraph value of load address
Data record	Value of code
End record	End of code

Each record consists of the following fields.

```
:   XX  XXXX  XX  DD......DD SS  NL
(1) (2) (3)   (4)     (5)    (6) (7)
```

Number	Description
(1)	Record mark
(2)	Number of bytes The number of bytes is expressed as 2-digit hexadecimal number of (5).
(3)	Location address
(4)	Record type 05: Start linear address record 04: Expanded linear address record 03: Start address record 02: Expanded address record 00: Data record 01: End record
(5)	Code Each byte of code is expressed as 2-digit hexadecimal number.
(6)	Checksum This is the 2-digit two's complement value of a result of hexadecimal addition of all bytes in the record except for ":", "SS", and "NL".
(7)	Newline (\n)

Remark The location address in the Intel HEX format is 2 bytes (16 bits).
Therefore, only a 64-Kbyte space can be directly specified.
To extend this area, the Intel HEX format adds the 16-bit expanded address so that a space of up to 1 Mbyte (20 bits) can be used.
Specifically, the record type that specifies the 16-bit expanded address is added.
This expanded address is shifted by four bits and added to the location address to express a 20-bit address.

3.5.2 Start linear address record

This indicates the linear address.

```
:   04  0000  05  XXXXXXXX  SS  NL
(1) (2) (3)   (4)    (5)    (6) (7)
```

Number	Description
(1)	Record mark
(2)	Fixed at 04
(3)	Fixed at 0000
(4)	Record type (Fixed at 05)
(5)	Linear address value
(6)	Checksum
(7)	Newline

3.5.3 Expanded linear address record

This indicates the upper 16-bit address at bits 32 to 16.

```
:   02  0000  04  XXXX  SS  NL
(1) (2) (3)   (4)  (5)  (6) (7)
```

Number	Description
(1)	Record mark
(2)	Fixed at 02
(3)	Fixed at 0000
(4)	Record type (Fixed at 04)
(5)	Upper 16-bit address at bits 32 to 16
(6)	Checksum
(7)	Newline

Note The location address of the data record is used as the lower 16 bits.

3.5.4 Start address record

This indicates the entry point address.

```
:   04  0000  03  PPPP  XXXX  SS  NL
(1) (2) (3)   (4)  (5)   (6)  (7) (8)
```

Number	Description
(1)	Record mark
(2)	Fixed at 04

Number	Description
(3)	Fixed at 0000
(4)	Record type (Fixed at 03)
(5)	Paragraph value of entry point address[Note]
(6)	Offset value of entry point address
(7)	Checksum
(8)	Newline

Note The address is calculated by (paragraph value << 4) + offset value.

3.5.5 Expanded address record

This indicates the paragraph value of the load address[Note].

Note This is output at the beginning of the segment (when the data record is output) or when the offset value of the data record's load address exceeds the maximum value of 0xffff and a new segment is output.

```
:   02  0000  02  PPPP  SS  NL
(1) (2) (3)   (4)  (5)  (6) (7)
```

Number	Description
(1)	Record mark
(2)	Fixed at 02
(3)	Fixed at 0000
(4)	Record type (Fixed at 02)
(5)	Paragraph value of segment
(6)	Checksum
(7)	Newline

3.5.6 Data record

This indicates the value of the code.

```
:   XX  XXXX  00  DD......DD  SS  NL
(1) (2) (3)   (4)     (5)     (6) (7)
```

Number	Description
(1)	Record mark
(2)	Number of bytes[Note]
(3)	Location address
(4)	Record type (Fixed at 00)

Number	Description
(5)	Code Each byte of code is expressed as 2-digit hexadecimal number.
(6)	Checksum
(7)	Newline

Note This is limited to the range of 0x1 to 0xff (the minimum value for the number of bytes of the code indicated by one data record is 1 and the maximum value is 255).

Example

```
:   04  0100  00  3C58E01B  6C  NL
(1) (2) (3)   (4)    (5)    (6) (7)
```

Number	Description
(1)	Record mark
(2)	Number of bytes of 3C58E01B expressed as 2-digit hexadecimal numbers
(3)	Location address
(4)	Record type 00
(5)	Each byte of code is expressed as 2-digit hexadecimal number.
(6)	Checksum The lower 1 byte of E6C, which is the two's complement of 04 + 01 + 00 + 00 + 3C + 58 + E0 + 1B = 194, is expressed as a 2-digit hexadecimal number.
(7)	Newline (\n)

3.5.7 End record

This indicates the end of the code.

```
:   00  0000  01  FF  NL
(1) (2) (3)   (4) (5) (6)
```

Number	Description
(1)	Record mark
(2)	Fixed at 00
(3)	Fixed at 0000
(4)	Record type (Fixed at 01)
(5)	Fixed at FF
(6)	Newline

3.6 Motorola S-record File

This section explains the Motorola S-record file.

The methods for outputting the Motorola S-record file are shown below.

(1) For CubeSuite+

Select the build tool node on the Project Tree panel and select the [Hex Output Options] tab on the Property panel.
To output the Motorola S-record file, select [Motorola S-record file(-FOrm=Stype)] in the [Hex file format] property in the [Hex Format] category.
Specify the output destination in the [Output folder] property and [Output file name] property in the [Output File] category.
It is also shown on the project tree, under the Build tool generated files node.

(2) Command line

When the -form=stype option is specified, the Motorola S-record file is output to the current folder under the output file name with the extension replaced by ".mot".

3.6.1 Structure of the Motorola S-record file

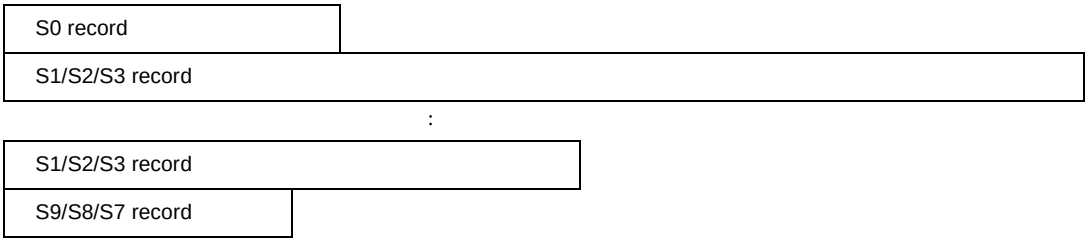
The Motorola S-record file consists of seven records[Note 1]: S0 record as the header record, S1, S2, and S3 records as the data record, and S9, S8, and S7 records as the end records[Note 2].

Notes 1. Each record is output in ASCII code.

2. The Motorola S-record files are divided into three types: 16-bit address type, (24-bit) standard address type, and 32-bit address type. The format of the 16-bit address type consists of S0, S1, and S9 records, the format of the standard address type consists of S0, S2, and S8 records, and the format of the 32-bit address type consists of S0, S3, and S7 records.

The structure and contents of the Motorola S-record file are shown below.

Figure 3-2. Structure of Motorola S-record File



Output Information	Description
S0 record	File name
S1 record	Value of code
S2 record	Value of code
S3 record	Value of code
S7 record	Entry point address
S8 record	Entry point address

Output Information	Description
S9 record	Entry point address

Each record consists of the following fields.

```
Sx  XX  YY......YY  SS  NL
(1) (2)     (3)     (4) (5)
```

Number	Description
(1)	Record type S0: S0 record S1: S1 record S2: S2 record S3: S3 record S4: S4 record S5: S5 record S6: S6 record S7: S7 record S8: S8 record S9: S9 record
(2)	Record length The number of bytes as 2-digit hexadecimal number of (3) + number of bytes expressed by "SS"[Note] is expressed.
(3)	Field
(4)	Checksum The one's complement is obtained from the sum of the number of 2-digit hexadecimal bytes in the record except for Sx, SS, and NL, and the lower one byte of the one's complement is expressed as a 2-digit hexadecimal number.
(5)	Newline (\n)

Note This is 1.

3.6.2 S0 record

This indicates the file name.

```
S0  0E  0000  XX......XX  SS  NL
(1) (2) (3)     (4)     (5) (6)
```

Number	Description
(1)	Fixed at S0
(2)	Fixed at 0E
(3)	Fixed at 0000
(4)	File name (eight characters) + file format (three characters) in most cases
(5)	Checksum
(6)	Newline

3.6.3 S1 record

This indicates the value of the code.

```
S1  XX  YYYY  ZZ......ZZ  SS  NL
(1) (2)  (3)      (4)     (5) (6)
```

Number	Description
(1)	Fixed at S1
(2)	Record length
(3)	Load address 16 bits (0x0 to 0xFFFF)
(4)	Code Each byte of code is expressed as 2-digit hexadecimal number.
(5)	Checksum
(6)	Newline

3.6.4 S2 record

This indicates the value of the code.

```
S2  XX  YYYYYY  ZZ......ZZ  SS  NL
(1) (2)   (3)       (4)     (5) (6)
```

Number	Description
(1)	Fixed at S2
(2)	Record length
(3)	Load address 24 bits (0x0 to 0xFFFFFF)
(4)	Code Each byte of code is expressed as 2-digit hexadecimal number.
(5)	Checksum
(6)	Newline

3.6.5 S3 record

This indicates the value of the code.

```
S3  XX  YYYYYYYY  ZZ......ZZ  SS  NL
(1) (2)   (3)         (4)     (5) (6)
```

Number	Description
(1)	Fixed at S3
(2)	Record length

Number	Description
(3)	Load address 32 bits (0x0 to 0xFFFFFFFF)
(4)	Code Each byte of code is expressed as 2-digit hexadecimal number.
(5)	Checksum
(6)	Newline

3.6.6 S7 record

This indicates the entry point address.

```
S7  XX  YYYYYYYY  SS  NL
(1) (2)   (3)     (4) (5)
```

Number	Description
(1)	Fixed at S7
(2)	Record length
(3)	Entry point address 32 bits (0x0 to 0xFFFFFFFF)
(4)	Checksum
(5)	Newline

3.6.7 S8 record

This indicates the entry point address.

```
S8  XX  YYYYYY  SS  NL
(1) (2)   (3)   (4) (5)
```

Number	Description
(1)	Fixed at S8
(2)	Record length
(3)	Entry point address 24 bits (0x0 to 0xFFFFFF)
(4)	Checksum
(5)	Newline

3.6.8 S9 record

This indicates the entry point address.

```
S9  XX  YYYY  SS  NL
(1) (2)  (3)  (4) (5)
```

Number	Description
(1)	Fixed at S9
(2)	Record length
(3)	Entry point address 16 bits (0x0 to 0xFFFF)
(4)	Checksum
(5)	Newline

APPENDIX A WINDOW REFERENCE

This appendix explains windows/panels/dialog boxes used in build process.

A.1 Description

The following lists the windows/panels/dialog boxes used in build process.

Table A-1. List of Windows/Panels/Dialog Boxes

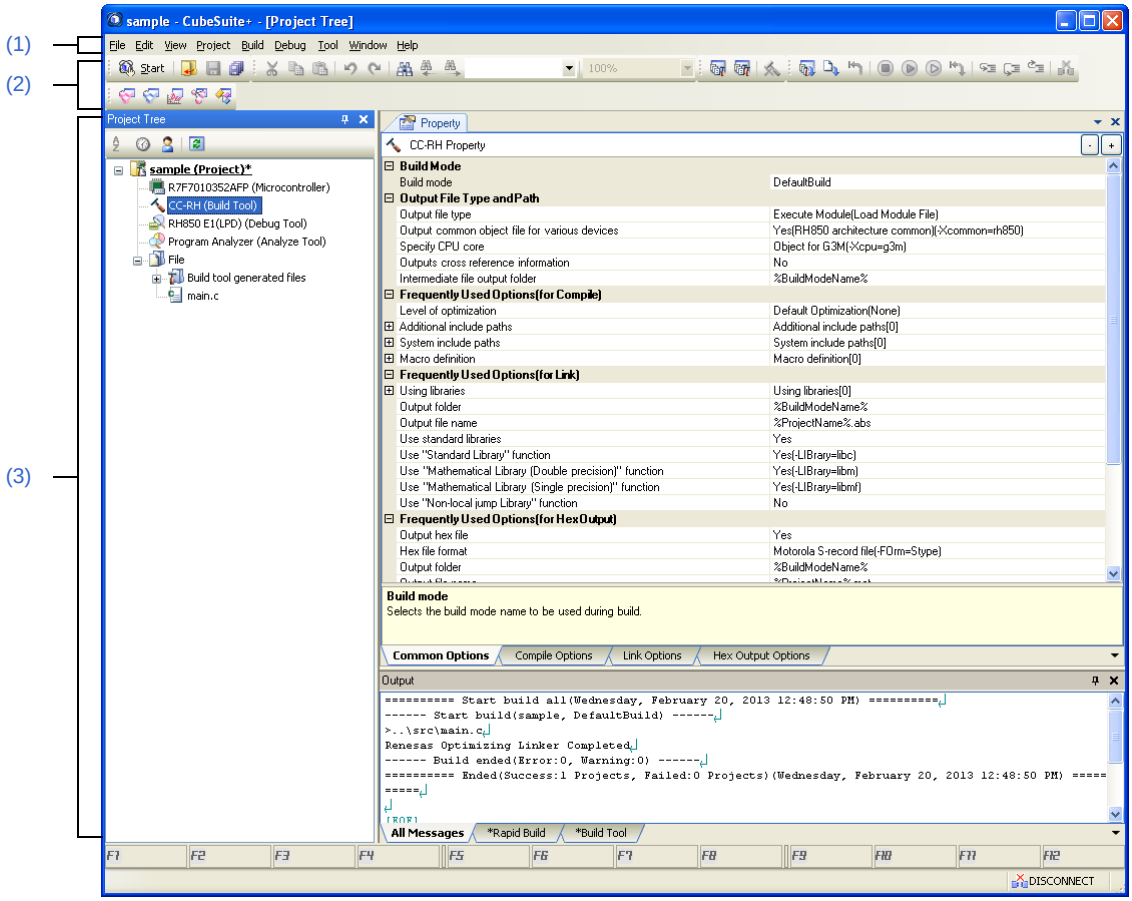
Window/Panel/Dialog Box Name	Function Description
Main window	This is the first window to be opened when CubeSuite+ is launched.
Project Tree panel	This panel is used to display the project components in tree view.
Property panel	This panel is used to display the detailed information on the Build tool node, file, or category node that is selected on the Project Tree panel and change the settings of the information.
Editor panel	This panel is used to display and edit text files and source files.
Output panel	This panel is used to display the message that is output from the build tool.
Add File dialog box	This dialog box is used to create a new file and add it to the project.
Add Folder and File dialog box	This dialog box is used to add existing files and folder hierarchies to the project.
Character String Input dialog box	This dialog box is used to input and edit characters in one line.
Text Edit dialog box	This dialog box is used to input and edit texts in multiple lines.
Path Edit dialog box	This dialog box is used to edit or add the path or file name including the path.
System Include Path Order dialog box	This dialog box is used to refer the system include paths specified for the compiler and set their specified sequence.
Specify Rule Number dialog box	This dialog box is used to select the number of the MISRA-C:2004 rule and set it to the area that this dialog box is called from.
Section Settings dialog box	This dialog box is used to add, modify, or delete sections.
Add Section dialog box Modify Section dialog box Add Overlay dialog box	These dialog boxes are used to set a section name when adding, modifying, or overlaying a section, respectively.
Section Address dialog box	This dialog box is used to set an address when adding or modifying a section.
Unassigned Section dialog box	This dialog box is used to delete sections.
Save Settings dialog box	This dialog box is used to set the encoding and newline code of the file that is editing on the Editor panel.
Link Order dialog box	This dialog box is used to refer files to be input to the linker and configure these link orders.
Build Mode Settings dialog box	This dialog box is used to add and delete build modes and configure the current build mode in batch.
Batch Build dialog box	This dialog box is used to do build, rebuild and clean process in batch with the build mode that the project has.
Progress Status dialog box	This dialog box is used to show how the process has been progressed.
Option dialog box	This dialog box is used to configure the CubeSuite+ environment.

Window/Panel/Dialog Box Name	Function Description
Add Existing File dialog box	This dialog box is used to select existing files to add to the project.
Import Build Options dialog box	This dialog box is used to select the target project file for importing the build options.
Browse For Folder dialog box	This dialog box is used to select the folder and set it to the area that this dialog box is called from.
Specify Far Jump File dialog box	This dialog box is used to select the Far Jump file and set it to the area that this dialog box is called from.
Specify MISRA-C:2004 Rule File dialog box	This dialog box is used to select the MISRA-C:2004 rule file and set it to the area that this dialog box is called from.
Add Excluding File dialog box	This dialog box is used to select the file that will not be checked against the MISRA-C:2004 rules and set it to the area that this dialog box is called from.
Specify Using Library File dialog box	This dialog box is used to select the library file and set it to the area that this dialog box is called from.
Save As dialog box	This dialog box is used to save the editing file or contents of each panel to a file with a name.
Open with Program dialog box	This dialog box is used to select the application to open the file.
Select Import File dialog box	This dialog box is used to select the file and import it to the area that this dialog box is called from.
Select Export File dialog box	This dialog box is used to output the information of the area that this dialog box is called from.

Main window

This is the first window to be opened when CubeSuite+ is launched.
This window is used to control the user program execution and open panels for the build process.

Figure A-1. Main Window



The following items are explained here.

- [How to open]
- [Description of each area]

[How to open]

- Select Windows [start] >> [All programs] >> [Renesas Electronics CubeSuite+] >> [CubeSuite+].

Remark In Windows 8, double-click on [CubeSuite+] on the start screen.

[Description of each area]

(1) Menu bar

The menus used in build process are displayed.

(a) [Project]

The [Project] menu shows menu items to operate the project and others.

Add New Subproject...	Closes the current project and opens the Create Project dialog box to create a new project. If the currently open project or file has been modified but it has not been saved yet, a confirmation message is displayed to ask you whether you want to save it.
Open Project...	Closes the current project and opens the Open Project dialog box to open the existing project. If the currently open project or file has been modified but it has not been saved yet, a confirmation message is displayed to ask you whether you want to save it.
Favorite Projects	Displays a cascading menu to use to open or save your favorite project.
1*path*	Opens your favorite project registered with [Favorite Projects] >> [1 Register to Favorite Project]. If no project has been registered, "Favorite Projects" is displayed.
2*path*	Opens your favorite project registered with [Favorite Projects] >> [2 Register to Favorite Project]. If no project has been registered, "Favorite Projects" is displayed.
3*path*	Opens your favorite project registered with [Favorite Projects] >> [3 Register to Favorite Project]. If no project has been registered, "Favorite Projects" is displayed.
4*path*	Opens your favorite project registered with [Favorite Projects] >> [4 Register to Favorite Project]. If no project has been registered, "Favorite Projects" is displayed.
1 Register to Favorite Project	The current project path is added to [1*path*] in [Favorite Projects].
2 Register to Favorite Project	The current project path is added to [2*path*] in [Favorite Projects].
3 Register to Favorite Project	The current project path is added to [3*path*] in [Favorite Projects].
4 Register to Favorite Project	The current project path is added to [4*path*] in [Favorite Projects].
Add	Displays the cascading menu to add subprojects to the project.
Add Subproject...	Opens the Add Existing Subproject dialog box to add an existing subproject to the project.
Add New Subproject...	Opens the Create Project dialog box to add a new subproject to the project.
Add File...	Opens the Add Existing File dialog box to add the selected file to the project.
Add New File...	Opens the Add File dialog box to create a file with the selected file type and add to the file to the project. The added file can be opened with the application corresponds to the file extension.
Add New Category	Adds a new category node to the root of the File node. This allows the category name to be changed. The default category name is "New category". You can also add a category node with the same name as an existing category node. Note that this menu is disabled when the build tool is in operation.

Sets *selected project or subproject* to an active project.	Sets the selected project or subproject to an active project.
Close Project	Closes the current project. If the currently open project or file has been modified but it has not been saved yet, a confirmation message is displayed to ask you whether you want to save it.
Save Project	Saves the setting information of the current project to the project file.
Save Project As...	Opens the Save Project As dialog box to save the setting information of the current project to the project file with another name.
Remove from Project	Removes the selected project or subproject from the project. The subproject files or the file themselves are not deleted from the file system.
Save Project and Development Tools as Package...	Saves a set of this product and the project by copying them in a folder. It is also possible to pack and save the project only.

(b) [Build]

The [Build] menu shows menu items for the build process and others.

Build Project	Runs a build of the project. A build of a subproject is also run when it is added in the project. Note that this menu is disabled when the build tool is in operation.
Rebuild Project	Runs a rebuild of the project. A rebuild of a subproject is also run when it is added in the project. Note that this menu is disabled when the build tool is in operation.
Clean Project	Runs a clean of the project. A clean of a subproject is also run when it is added in the project. Note that this menu is disabled when the build tool is in operation.
Rapid build	Toggles the rapid build function between enabled (default) and disabled.
Update Dependencies	Updates the dependency of a file in the project to build. The dependency of a file in the subproject to be build is also updated when the subproject is added to the project.
Build *active project*	Runs a build of the active project. If the active project is the main project, a build of its subproject is not run. Note that this menu is disabled when the build tool is in operation.
Rebuild *active project*	Runs a rebuild of the active project. If the active project is the main project, a rebuild of its subproject is not run. Note that this menu is disabled when the build tool is in operation.
Clean *active project*	Runs a clean of the active project. If the active project is the main project, a clean of its subproject is not run. Note that this menu is disabled when the build tool is in operation.
Update Dependencies of *active project*	Updates the dependency of a file in the active project to build.
Stop Build	Cancels the build, rebuild, batch build and clean operation.
Build Mode Settings...	Opens the Build Mode Settings dialog box to modify and add to the build mode.
Batch Build...	Opens the Batch Build dialog box to run a batch build.
Build Option List	Lists the currently set build options in the Output panel.

(2) Toolbar

The buttons used in build process are displayed.

(a) Build toolbar

Build toolbar shows buttons used in build process.

	Runs a build of the project. A build of a subproject is also run when it is added in the project. Note that this button is disabled when the build tool is in operation.
	Runs a rebuild of the project. A rebuild of a subproject is also run when it is added in the project. Note that this button is disabled when the build tool is in operation.
	Cancels the build, rebuild, batch build and clean operation.

(3) Panel display area

The following panels are displayed in this area.

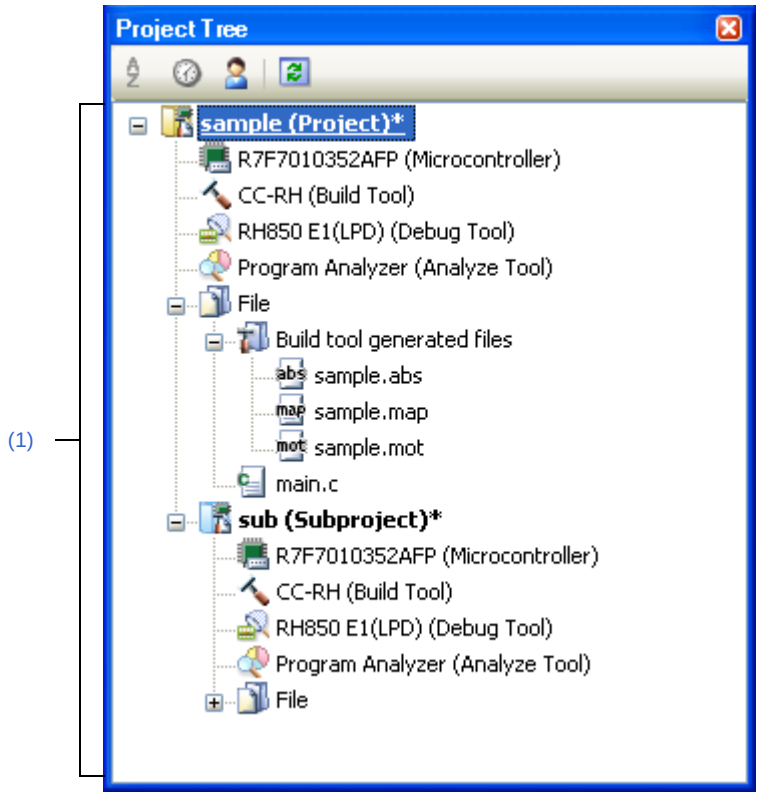
- Project Tree panel
- Property panel
- Editor panel
- Output panel

See the each panel section for details of the contents of the display.

Project Tree panel

This panel is used to displays the project components such as the build tool, source files, etc. in tree view.

Figure A-2. Project Tree Panel



The following items are explained here.

- [How to open]
- [Description of each area]
- [[Edit] menu (only available for the Project Tree panel)]
- [Context menu]

[How to open]

- From the [View] menu, select [Project Tree].

[Description of each area]

(1) Project tree area

This area displays the project components in tree view with the following given nodes.

Node	Description
Project name (Project) (hereafter referred to as "Project node")	The project name.
Build tool name (Build tool) (hereafter referred to as "Build tool node")	The build tool (CC-RH) used in the project.
File (hereafter referred to as "File node")	Files that are added to the project are displayed directly below this node.
Build tool generated files (hereafter referred to as "Build tool generated files node")	The following files generated by the build tool appear directly below the node generated during the build. - For other than library projects Load module file (*.abs) Intel Hex file (*.hex) Motorola S-record file (*.mot) Binary file (*.bin) Link map file (*.map) Library file (*.lib) Library list file (*.lbp) Stack information file (*.sni) Error message file (*.err) - For library projects Library file (*.lib) Library list file (*.lbp) Error message file (*.err) Files displayed below this node cannot be renamed, deleted, or moved. This node is always generated below the File node. This node will no longer appear if you reload the project after building.
Category name (hereafter referred to as "category node")	Categories that the user creates to categorize files (see "2.3.4 Classify a file into a category"). This node is always created below the File node.
Subproject name (Subproject) (hereafter referred to as "Subproject node")	Subprojects added to the project.

When each component (the node or file) is selected, the detailed information (property) is displayed on the Property panel. You can change the settings.

Remark When multiple components are selected, only the tab that is common to all the components is displayed.
When multiple files are selected and the values of their common properties are different, then the corresponding value fields are displayed blank.

This area has the following functions.

(a) Adding a file

You can add a file by one of the following procedures.
The files are added below the File node.

<1> Adding an existing file

- Select either one of the Project node, Subproject node, File node or a file. Then select [Add] >> [Add File...] from the [File] menu. The Add Existing File dialog box will open. Select the file to be added.
- Select either one of the Project node, Subproject node, File node. Then select [Add] >> [Add File...] from the context menu. The Add Existing File dialog box will open. Select the file to be added.
- Copy the file using such as Explorer and then point the mouse to this area. Select [Paste] from the [Edit] menu.
- Drag a file using such as Explorer and drop it onto the location in this area where you want to add it to.

Remark If you drag the file using such as Explorer and drop it onto the empty space below the lower project tree, it is assumed that you have dropped it onto the Main project.

<2> Adding a new file

- Select either one of the Project node, Subproject node, File node or a file. Then select [Add] >> [Add New File...] from the [File] menu. The Add File dialog box will open. Designate the file to be created.
- Select either one of the Project node, Subproject node, File node. Then select [Add] >> [Add New File...] from the context menu. The Add File dialog box will open. Designate the file to be created.

Remark A blank file is created at the location designated in the Add File dialog box.

(b) Removing a file from the project

You can remove a file from the project by one of the following procedures.
The file itself is not deleted from the file system.

- Select the file you want to remove from the project and then select [Remove from Project] from the [Project] menu.
- Select the file you want to remove from the project and then select [Remove from Project] from the context menu.

(c) Moving a file

You can move a file by the following procedure.
The file is moved below the File node.

- Drag the file you want to move and then drop it onto the destination.

Remarks 1. If you drop the file in the main project or subproject, the individual options set in the file are retained.

2. If you drop the file between the different projects, or onto the main project or subproject in same project, the file is copied, not moved.
Note that the individual option set in the file are not retained.

(d) Adding a category

You can add a category node by one of the following procedures.
The category node is added below the File node.

- Select [Add New Category] from the [Project] menu.
- Select [Add New Category] from the context menu of either one of the Project node, Subproject node, or File node.

Remarks 1. The default category name is "New category".

2. You can also add a category node with the same name as the existing category node.

(e) Moving a category

You can move a category node by the following procedure.
The category node is moved below the File node.

- Drag the category node you want to move and then drop it onto the destination.

Remarks 1. If you drop the category node in the main project or subproject, the individual options set in the file in the category node are retained.

2. If you drop the category node between the different projects, or onto the main project or subproject in same project, the category node is copied, not moved.
Note that the individual option set in the file in the category node are not retained.

(f) Adding a folder

You can add a folder using such as Explorer by the following procedure.
The folder is added below the File node.
The folder is added as a category.

- Drag the folder using such as Explorer and drop it onto the destination. The Add Folder and File dialog box will open. Specify the file types and subfolder levels in the folder to be added.

Caution You cannot drag and drop a folder and file onto this area simultaneously.

(g) Editing the display of the subproject build order

Subprojects are displayed in order of build from the top.
Therefore, the build order can be changed by changing the display order of subprojects.
Builds of the project are run from the subprojects then the main project.

(h) Setting the standard build options

On the Property panel, if you add a change to the settings for the standard build options, the value of the property will be displayed in boldface.
You can change the current build options to the standard setting (cancel boldface) by the following procedure.

- Select the Build tool node and then select [Set to Default Build Option for Project] in the context menu.

Remark The setting of the standard build options takes effect to the whole project (main project and subprojects).

(i) Sorting files and categories

You can sort files and category nodes in order of the file name, time stamp, or user definition by the following procedure.

- Select one of the buttons on the toolbar.

The description of each button is shown below.
is selected by default.

Button	Description
	Sorts category nodes and files in order of their names. : Ascending order : Descending order : Ascending order
	Sorts category nodes and files in order of their timestamp. : Descending order : Ascending order : Descending order
	Displays category nodes and files (exclude the dependency files) in order of the user definition (default). You can change the display order of category nodes and files arbitrarily by dragging and dropping them.

(j) Displaying the dependency file

When there is a dependency file for a source file added to the project, the dependency file is displayed under the source file.

The source file without a dependency file	main.c
The source file with a dependency file	main.c io.h

The display of the dependency files is updated on the following timings:

- When the first build is run after the project is loaded
- When on the toolbar is clicked
- When [Update Dependencies] is selected from the [Build] menu
- When [Update Dependencies of *active project*] is selected from the [Build] menu

Remarks 1. This function is valid only when the [Show dependency files in project tree] checkbox in the [General - Build/Debug] category of the Option dialog box is selected.

2. Information on the dependency files displayed on the project tree is not saved in the project file.

(k) Displaying the editing file

When the file added to the project is edited in the Editor panel (exclude the dependency files) and the file is not saved once, the file name is followed by "*".
If the file is saved, "*" will be deleted.

The file that is saved	main.c
The file that is not saved after editing	main.c*

(l) Displaying the source file in boldface that the individual build option is set

The source file icon whose option is different from the project general options (individual compile options or individual assemble options) is changed to the different one from the normal icon.

The file with project general options	main.c
The file with the individual build options	main.c

(m) Highlighting the file with the read-only attribute

The read-only file added to the project is displayed in italic.

The file without the read-only attribute	main.c
The file with the read-only attribute	*main.c*

(n) Highlighting the file that does not exist

The file that is added to the project but does not exist is grayed out and its icon is dimmed.

The file that exists	main.c
The file that does not exist	*main.c*

(o) Highlighting the build-target file

<1> The file which an error occurred during building (rapid building), rebuilding, compiling or assembling is highlighted as the example below.

The file without errors or warnings	main.c
The file with an error	main.c
The file with a warning	main.c

Remarks 1. The file with both an error and a warning is highlighted in red.

2. The highlight is canceled when the build option (general option or individual option) or the build mode is changed.

<2> The names of the following files are displayed in boldface.

- The source file that has not been compiled after editing
- The source file after a clean has been executed
- The source file after the build tool option has been changed
- The source file after the build mode has been changed

Remark The file names are all displayed in boldface right after the project is opened. The boldface display is canceled after a build is executed.

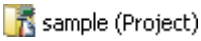
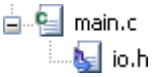
(p) Highlighting the non build-target file

The file that is set as non build-target is highlighted as shown in the example below.

Build-target file	main.c
Non build-target file	main.c

(q) Highlighting overlay icons

Overlay icons of Windows Explorer set in a project, a file added to the project, and a category (only when a shortcut to the folder is set) is displayed on the left side of the ordinal icon as the example below.

The project that has not been changed	
The project displaying an overlay icon	

Caution **The above overlay icon is provided as a sample.**
Note that an icon to be displayed differs depending on the tools in use.

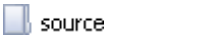
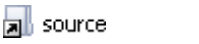
The display of overlay icons is updated on the following timings:

- When the project is loaded
- When on the toolbar is clicked
- When [Refresh] is selected from the [Edit] menu

Remark This function is valid only when this product is activated while the [Show overlay icons for Windows Explorer in project tree] checkbox in the [General - Display] category of the Option dialog box is selected.

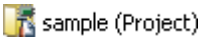
(r) Highlighting the category in which a shortcut to a folder is set

The category in which a shortcut to a folder is set is highlighted as shown in the example below.

Ordinal category	
Category in which a shortcut to a folder has been set	

(s) Highlighting the project that has been changed

The file component that is added to the project and the property of the project component are changed, the project name is followed by "*" and is displayed in boldface.
The boldface is canceled when the project is saved.

The project that has not been changed	
The project that has been changed	

(t) Highlighting the active project

The active project is underlined.

Non-active project	
Active project	

(u) Updating the status of file highlighting

The state of highlighting for files, read-only files, non-existent files, and overlay icons is updated by the following procedure.

- Select on the toolbar.

(v) Running the editor

The file with the specific extension is opened in the Editor panel. When an external text editor is set to be used in the Option dialog box, the file is opened with the external text editor that has been set. Other files are opened with the applications associated by the host OS.

Caution The file with the extension that is not associated with the host OS is not displayed.

You can open the editor by one of the following procedures.

- Double click the file.
- Select a file and then select [Open] from the context menu.
- Select a file and then press the [Enter] key.

The files that can be opened with the Editor panel are shown below.

- C source file (*.c)
- Assembly source file (*.asm, *.s)
- Symbol address file (*.fsy)
- Header file (*.h, *.inc)
- List file for assembly program (*.lst, *.lis)
- Preprocessor expanded file for assembly program (*.exp)
- Assemble list file (*.prn)
- Link order specification file (*.mtls)
- Link map file (*.map)
- Library list file (*.lbp)
- Intel Hex file (*.hex)
- Motorola S-record file (*.mot)
- Text file (*.txt)

Remark You can use one of the procedures below to open files other than those listed above in the Editor panel.

- Drag a file and drop it onto the Editor panel.
- Select a file and then select [Open with Internal Editor...] from the context menu.

[[Edit] menu (only available for the Project Tree panel)]

Copy	Copies the selected file or category node to the clipboard. When the file name or category name is in editing, the characters of the selection are copied to the clipboard. Note that this menu is enabled when a file (exclude the dependency files) or category is selected.
Paste	Inserts the contents of the clipboard directly below the selected node on the project tree. When the file name or category name is in editing, the contents of the clipboard are inserted. Note that this menu is disabled when the contents of the clipboard exist in the same project, when multiple files and category nodes are selected, and when the build tool is in operation.
Rename	You can rename the selected project, subproject, file, or category node.Press the [Enter] key to confirm editing. Press the [ESC] key to cancel editing. When the file is selected, the actual file name is also changed. When the selected file is added to other project, that file name is also changed. Note that this menu is enabled when a project, subproject, file (exclude the dependency files), or category node is selected. Note that this menu is disabled when the build tool is in operation.

[Context menu]

(1) When the Project node is selected

Build *active project*	Runs a build of the active project. If the active project is the main project, a build of its subproject is not run. Note that this menu is disabled when the build tool is in operation.
Rebuild *active project*	Runs a rebuild of the active project. If the active project is the main project, a rebuild of its subproject is not run. Note that this menu is disabled when the build tool is in operation.
Clean *active project*	Runs a clean of the active project. If the active project is the main project, a clean of its subproject is not run. Note that this menu is disabled when the build tool is in operation.
Open Folder with Explorer	Opens the folder that contains the project file of the selected project with Explorer.
Windows Explorer Menu	Displays the context menu in Windows Explorer corresponding to the project file of the selected project.
Add	Displays the cascading menu to add subprojects and files to the project.
Add Subproject...	Opens the Add Existing Subproject dialog box to add the selected subproject to the project.
Add New Subproject...	Opens the Create Project dialog box to add the created subproject to the project.
Add File...	Opens the Add Existing File dialog box to add the selected file to the project.
Add New File...	Opens the Add File dialog box to create a file with the selected file type and add to the file to the project. The added file can be opened with the application corresponds to the file extension.
Add New Category	Adds a category node directly below the File node. This allows the category name to be changed. Up to 200 characters can be specified. The default category name is "New category". You can also add a category node with the same name as the existing category node. This menu is disabled when the build tool is in operation and when categories are nested 20 levels.
Set *selected project* as Active Project	Sets the selected project to the active project.
Save Project and Development Tools as Package...	Saves a set of this product and the project by copying them in a folder. It is also possible to pack and save the project only.
Paste	This menu is always disabled.
Rename	You can rename the selected project.
Property	Displays the selected project's property on the Property panel.

(2) When the Subproject node is selected

Build *active project*	Runs a build of the active project. Note that this menu is disabled when the build tool is in operation.
Rebuild *active project*	Runs a rebuild of the active project. Note that this menu is disabled when the build tool is in operation.

Clean *active project*	Runs a clean of the active project. Note that this menu is disabled when the build tool is in operation.
Open Folder with Explorer	Opens the folder that contains the subproject file of the selected subproject with Explorer.
Windows Explorer Menu	Displays the context menu in Windows Explorer corresponding to the subproject file of the selected subproject.
Add	Displays the cascading menu to add subprojects, files, and category nodes to the project.
Add Subproject...	Opens the Add Existing Subproject dialog box to add the selected subproject to the project. The subproject cannot be added to another subproject.
Add New Subproject...	Opens the Create Project dialog box to add the created subproject to the project. The subproject cannot be added to another subproject.
Add File...	Opens the Add Existing File dialog box to add the selected file to the project.
Add New File...	Opens the Add File dialog box to create a file with the selected file type and add to the file to the project. The added file can be opened with the application corresponds to the file extension.
Add New Category	Adds a category node directly below the File node. This allows the category name to be changed. Up to 200 characters can be specified. The default category name is "New category". You can also add a category node with the same name as the existing category node. This menu is disabled when the build tool is in operation and when categories are nested 20 levels.
Set *selected subproject* as Active Project	Sets the selected subproject to the active project.
Remove from Project	Removes the selected subproject from the project. The subproject file itself is not deleted from the file system. When the selected subproject is the active project, it cannot be removed from the project. Note that this menu is disabled when the build tool is in operation.
Paste	This menu is always disabled.
Rename	You can rename the selected subproject.
Property	Displays the selected subproject's property on the Property panel.

(3) When the Build tool node is selected

Build Project	Runs a build the selected project (main project or subproject). A build of a subproject is also run when it is added in the project. Note that this menu is disabled when the build tool is in operation.
Rebuild Project	Runs a rebuild the selected project (main project or subproject). A rebuild of a subproject is also run when it is added in the project. Note that this menu is disabled when the build tool is in operation.
Clean Project	Runs a clean of the selected project (main project or subproject). A clean of a subproject is also run when it is added in the project. Note that this menu is disabled when the build tool is in operation.

Set to Default Build Option for Project	Sets the current build options as the standard for the project. When a subproject is added, its setting is not made. When the build option that is different from the standard option is set, its property is displayed in boldface.
Import Build Options...	Opens the Import Build Options dialog box to import the build options from the selected project file.[Note]
Set Link Order...	Opens the Link Order dialog box to display object module files and library files to and set their link orders. Note that this menu is disabled when the build tool is in operation.
Property	Displays the selected build tool's property on the Property panel.

Note See "2.11.1 Import the build options of other project" for details about the import function of the build options.

(4) When the Files node is selected

Add	Shows the cascading menu to add files and category nodes to the project.
Add File...	Opens the Add Existing File dialog box to add the selected file to the project. The file is added directly below this node. The added file can be opened with the application corresponds to the file extension.
Add New File...	Opens the Add File dialog box to create a file with the selected file type and add to the file to the project. The file is added directly below this node. The added file can be opened with the application corresponds to the file extension.
Add New Category	Adds a category node directly below this node. This allows the category name to be changed. Up to 200 characters can be specified. The default category name is "New category". You can also add a category node with the same name as the existing category node. This menu is disabled when the build tool is in operation and when categories are nested 20 levels.
Open Folder with Explorer	This menu is always disabled.
Windows Explorer Menu	This menu is always disabled.
Remove from Project	This menu is always disabled.
Copy	This menu is always disabled.
Paste	Inserts the contents of the clipboard directly below this node. However, this menu is disabled when the contents of the clipboard exist in the same project.
Rename	This menu is always disabled.
Property	Displays this node's property on the Property panel.

(5) When a file is selected

Compile	Compiles the selected C source file. Note that this menu is only displayed when a C source file (except a non build-target file) is selected. Note that this menu is disabled when the build tool is in operation.

Assemble	Assembles the selected assembly source file. Note that this menu is only displayed when an assembly source file (except a non build-target file) is selected. Note that this menu is disabled when the build tool is in operation.
Open	Opens the selected file with the application corresponds to the file extension (see "(v) Running the editor").
Open with Internal Editor...	Opens the selected file with the Editor panel.
Open with Selected Application...	Opens the Open with Program dialog box to open the selected file with the designated application.
Open Folder with Explorer	Opens the folder that contains the selected file with Explorer.
Windows Explorer Menu	Displays the context menu in Windows Explorer corresponding to the selected file.
Add	Shows the cascading menu to add files and category nodes to the project.
Add File...	Opens the Add Existing File dialog box to add the selected file to the project. The file is added to the same level as the selected file.
Add New File...	Opens the Add File dialog box to create a file with the selected file type and add to the file to the project. The file is added to the same level as the selected file. The added file can be opened with the application corresponds to the file extension.
Add New Category	Adds a new category node at the same level as the selected file. You can rename the category. Up to 200 characters can be specified. The default category name is "New category". You can also add a category node with the same name as the existing category node. This menu is disabled when the build tool is in operation and when categories are nested 20 levels.
Remove from Project	Removes the selected file from the project. The file itself is not deleted from the file system. Note that this menu is disabled when the build tool is in operation.
Copy	Copies the selected file to the clipboard. When the file name is in editing, the selected characters are copied to the clipboard.
Paste	This menu is always disabled.
Rename	You can rename the selected file. The actual file is also renamed. When the selected file is added to other project, that file name is also changed.
Change Extension...	Opens a message dialog box to confirm whether to change the file extension. Clicking on the [Yes] button in the dialog box will open the Character String Input dialog box, in which the extension of the selected file can be changed. When multiple files are selected, they are changed at one time. Note that this menu is disabled when a file that cannot be renamed or removed from the project is selected, and when the build tool is in operation.
Property	Displays the selected file's property on the Property panel.

(6) When the Build tool generated files node is selected

Property	Displays this node's property on the Property panel.

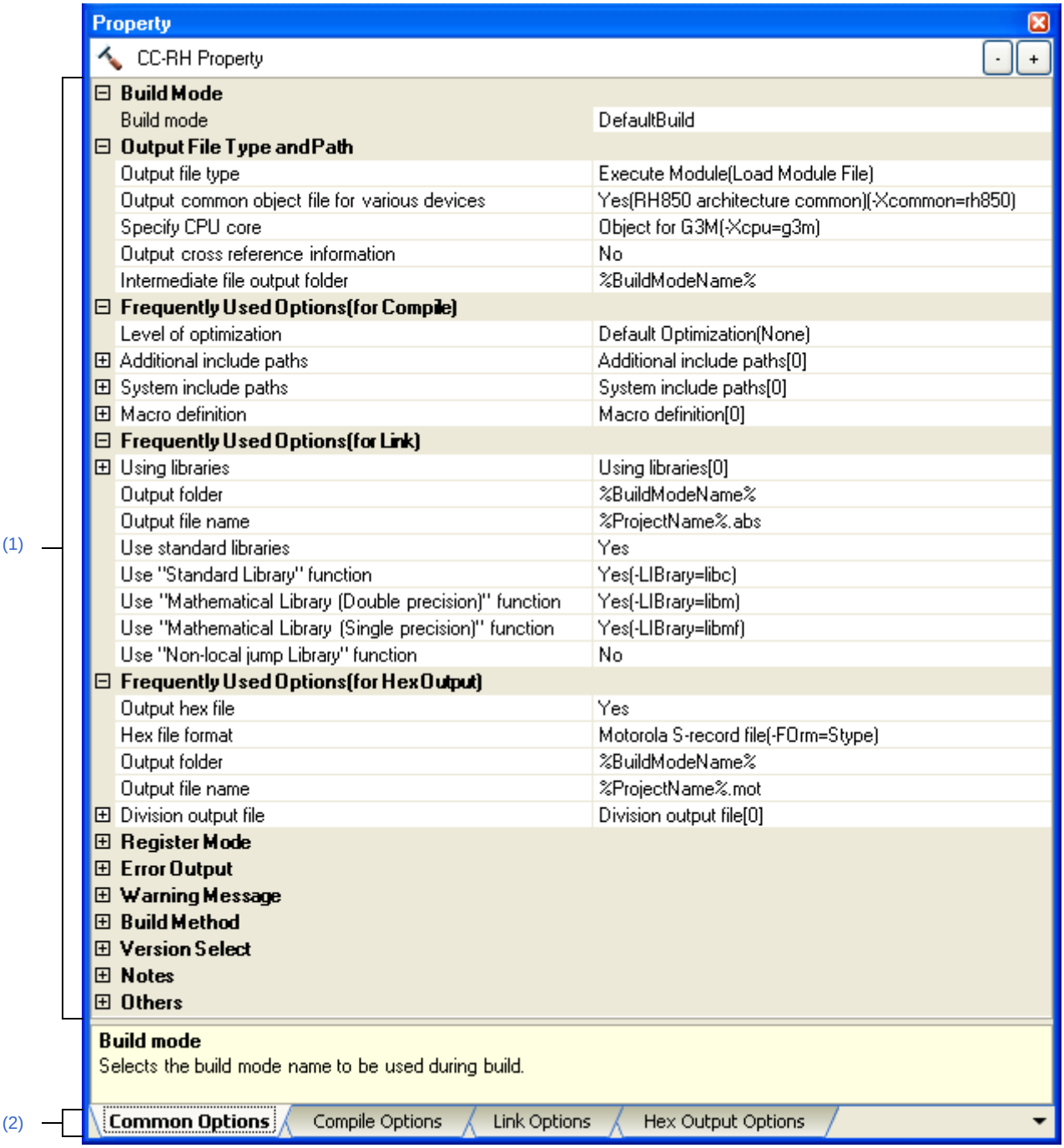
(7) When a category node is selected

Add	Shows the cascading menu to add files and category nodes to the project.
Add File...	Opens the Add Existing File dialog box to add the selected file to the project. The file is added directly below this node. The added file can be opened with the application corresponds to the file extension.
Add New File...	Opens the Add File dialog box to create a file with the selected file type and add to the file to the project. The file is added directly below this node. The added file can be opened with the application corresponds to the file extension.
Add New Category	Adds a category node directly below this node. This allows the category name to be changed. Up to 200 characters can be specified.f The default category name is "New category". You can also add a category node with the same name as the existing category node. This menu is disabled when the build tool is in operation and when categories are nested 20 levels.
Open Folder with Explorer	Opens the shortcut destination to the folder set in the selected category by Explorer. Note that this menu is disabled when the shortcut to the folder is not set.
Windows Explorer Menu	Displays the context menu in Windows Explorer corresponding to the shortcut destination in the folder set in the selected category. Note that this menu is disabled when the shortcut to the folder is not set.
Remove from Project	Removes the selected category node from the project. Note that this menu is disabled when the build tool is in operation.
Copy	Copies the selected category node to the clipboard. When the category name is in editing, the characters of the selection are copied to the clipboard.
Paste	Inserts the contents of the clipboard directly below this node. However, this menu is disabled when the contents of the clipboard exist in the same project. When the category name is in editing, the contents of the clipboard are inserted.
Rename	You can rename the selected category node.
Property	Displays the selected category node's property on the Property panel.

Property panel

This panel is used to display the detailed information on the Build tool node, file, or category node that is selected on the Project Tree panel by every category and change the settings of the information.

Figure A-3. Property Panel



The following items are explained here.

- [How to open]
- [Description of each area]
- [[Edit] menu (only available for the Property panel)]
- [Context menu]

[How to open]

- On the Project Tree panel, select the Build tool node, file, or category node, and then select [Property] from the [View] menu or [Property] from the context menu.

Remark When either one of the Build tool node, file, or category node on the Project Tree panel is selected while the Property panel has been opened, the detailed information of the selected item is displayed.

[Description of each area]

(1) Detailed information display/change area

In this area, the detailed information on the Build tool node, file, or category node that is selected on the Project Tree panel is displayed by every category in the list. And the settings of the information can be changed directly. Mark ⊟ indicates that all the items in the category are expanded. Mark ⊞ indicates that all the items are collapsed. You can expand/collapse the items by clicking these marks or double clicking the category name. Mark HEX indicates that only a hexadecimal number is allowed to input in the text box.
See the section on each tab for the details of the display/setting in the category and its contents.

(2) Tab selection area

Categories for the display of the detailed information are changed by selecting a tab.
In this panel, the following tabs are contained (see the section on each tab for the details of the display/setting on the tab).

(a) When the Build tool node is selected on the Project Tree panel

- [Common Options] tab
- [Compile Options] tab
- [Assemble Options] tab
- [Link Options] tab
- [Hex Output Options] tab
- [Create Library Options] tab

(b) When a file is selected on the Project Tree panel

- [Build Settings] tab (for C source file, assembly source file, object file, and libraryfile)
- [Individual Compile Options] tab (for C source file)
- [Individual Assemble Options] tab (for assembly source file)
- [File Information] tab

(c) When the category node, Files node, or Build tool generated files node is selected on the Project Tree panel

- [Category Information] tab

Remark When multiple components are selected on the Project Tree panel, only the tab that is common to all the components is displayed.
If the value of the property is modified, that is taken effect to the selected components all of which are common to all.

[[Edit] menu (only available for the Property panel)]

Undo	Cancels the previous edit operation of the value of the property.
Cut	While editing the value of the property, cuts the selected characters and copies them to the clipboard.
Copy	Copies the selected characters of the property to the clipboard.
Paste	While editing the value of the property, inserts the contents of the clipboard.
Delete	While editing the value of the property, deletes the selected characters.
Select All	While editing the value of the property, selects all the characters of the selected property.

[Context menu]

Undo	Cancels the previous edit operation of the value of the property.
Cut	While editing the value of the property, cuts the selected characters and copies them to the clipboard.
Copy	Copies the selected characters of the property to the clipboard.
Paste	While editing the value of the property, inserts the contents of the clipboard.
Delete	While editing the value of the property, deletes the selected characters.
Select All	While editing the value of the property, selects all the characters of the selected property.
Reset to Default	Restores the configuration of the selected item to the default configuration of the project. For the [Individual Compile Options] tab and [Individual Assemble Options] tab, restores to the configuration of the general option.
Reset All to Default	Restores all the configuration of the current tab to the default configuration of the project. For the [Individual Compile Options] tab and [Individual Assemble Options] tab, restores to the configuration of the general option.

[Common Options] tab

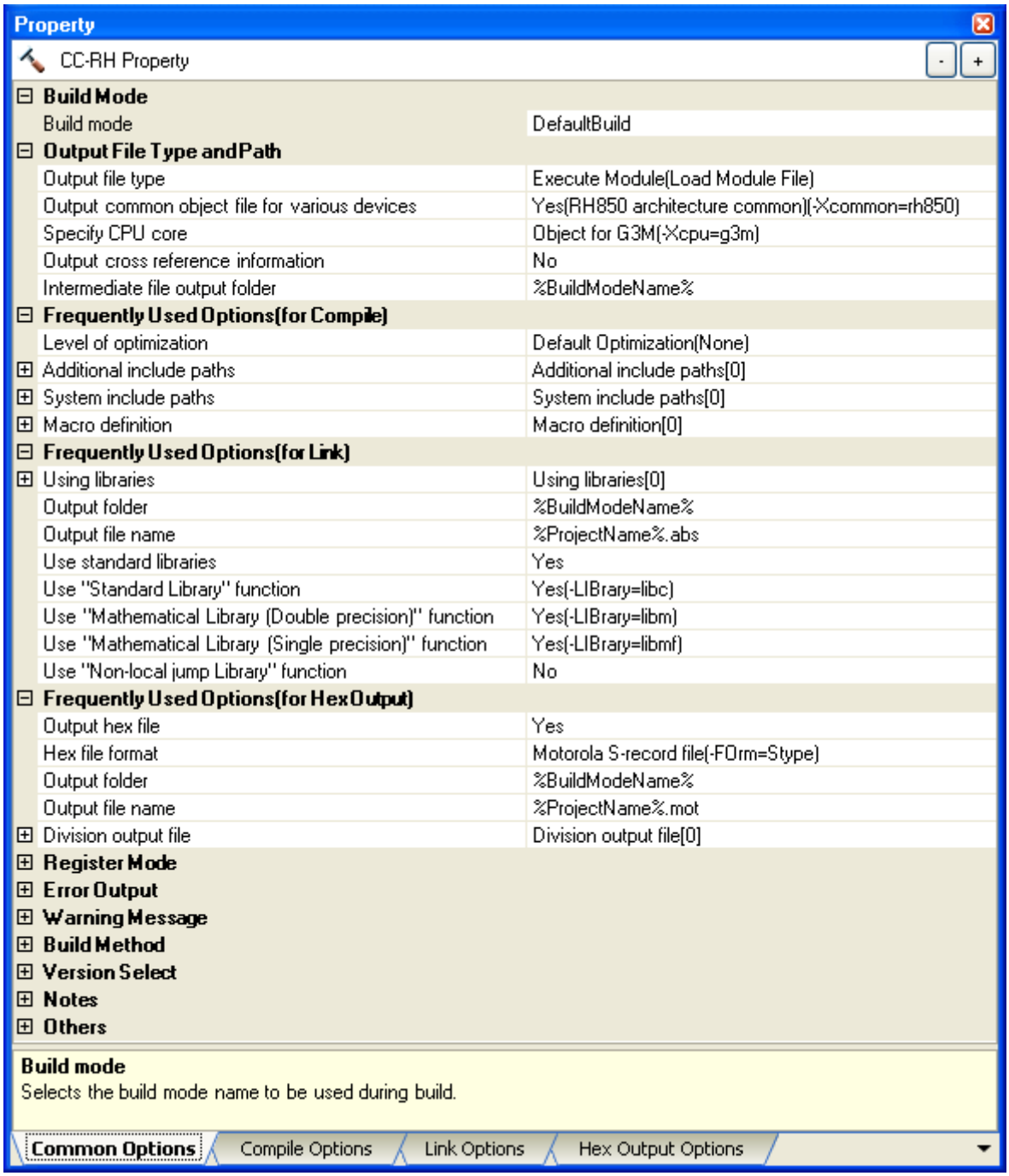
This tab shows the detailed information on the build tool categorized by the following and the configuration can be changed.

(1) [Build Mode]
(2) [Output File Type and Path]
(3) [Frequently Used Options(for Compile)]
(4) [Frequently Used Options(for Assemble)]
(5) [Frequently Used Options(for Link)]
(6) [Frequently Used Options(for Hex Output)]
(7) [Frequently Used Options(for Create Library)]
(8) [Register Mode]
(9) [Error Output]
(10) [Warning Message]
(11) [Build Method]
(12) [Version Select]
(13) [Notes]
(14) [Others]

Remark If the property in the [Frequently Used Options] category is changed, the value of the property having the same name contained in the corresponding tab will be changed accordingly.

Category from [Common Options] Tab	Corresponding Tab
[Frequently Used Options(for Compile)] category	[Compile Options] tab
[Frequently Used Options(for Assemble)] category	[Assemble Options] tab
[Frequently Used Options(for Link)] category	[Link Options] tab
[Frequently Used Options(for Hex Output)] category	[Hex Output Options] tab
[Frequently Used Options(for Create Library)] category	[Create Library Options] tab

Figure A-4. Property Panel: [Common Options] Tab



[Description of each category]

(1) [Build Mode]

The detailed information on the build mode is displayed and the configuration can be changed.

Build mode	Select the build mode to be used during a build. Note that this property is not applied to [Reset All to Default] from the context menu.		
	Default	DefaultBuild	
	How to change	Select from the drop-down list.	
	Restriction	DefaultBuild	Runs a build with the default build mode that is set when a new project is created.
		Build mode that is added to the project	Runs a build with the build mode that is added to the project (other than DefaultBuild).

(2) [Output File Type and Path]

The detailed information on output file types and paths is displayed and the configuration can be changed.

Output file type	Select the type of the file to be generated during a build. The file type set here will be the debug target. For other than library projects, only [Execute Module(Load Module File)] and [Execute Module(Hex File)] are displayed. For the library project, only [Library] is displayed.		
	Default	- For other than the library project Execute Module(Load Module File) - For the library project Library	
	How to change	Select from the drop-down list.	
	Restriction	Execute Module(Load Module File)	Generates a load module file during a build. The load module file will be the debug target.
		Execute Module(Hex File)	Generates a hex file during a build. The hex file will be the debug target. This item is displayed only when [Yes] in the [Output hex file] property in the [Output File] category from the [Hex Output Options] tab is selected.
		Library	Generates a library file during a build.
Output common object file for various devices	Specify the output of an object file common to the various devices. This corresponds to the -Xcommon option of the ccrh command.		
	Default	Yes(RH850 architecture common)(-Xcommon=rh850)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(RH850 architecture common)(-Xcommon=rh850)	The result is the same as when [Yes(V850E3V5 architecture common)(-Xcommon=v850e3v5)] is specified.
		Yes(V850E3V5 architecture common)(-Xcommon=v850e3v5)	Outputs the object file compatible with models having the V850E3V5 instruction set architecture.

Specify CPU core	Specify the CPU core. This corresponds to the -Xcpu option of the ccrh command.		
	Default	Object for G3M(-Xcpu=g3m)	
	How to change	Select from the drop-down list.	
	Restriction	Object for G3M(-Xcpu=g3m)	Generates an object for G3M.
		Object for G3K(-Xcpu=g3k)	Generates an object for G3K.
Output cross reference information	Select whether to output the cross reference information to a file. The file is output to the folder specified in the [Output folder] property in the [Output File] category from the [Link Options] tab. The file is output under the file name specified in the [Output file name] property with the extension replaced by ".cref". However, if the [Object file name] property in the [Output File] category from the [Individual Compile Options] tab is specified, the file is output under the file name specified in the property with the extension replaced by ".cref". This property is changed to [Yes(-Xcref)] when [Yes] in the [Compulsorily output cross reference file] property of the analyze tool is selected. If this property is changed to [No] when [Yes] in the [Compulsorily output cross reference file] property is selected, this property will be changed to [Yes(-Xcref)] during a build. This corresponds to the -Xcref option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xcref)	Outputs the cross reference information. The build processing speed slows down, but "jump to function" can be used.
		No	Does not output the cross reference information.
Intermediate file output folder	Specify the folder which the intermediate file is output. If a relative path is specified, the reference point of the path is the main project or subproject folder. If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different). The following placeholder is supported. %BuildModeName%: Replaces with the build mode name. If this is blank, it is assumed that the project folder has been specified. This corresponds to the -Xobj_path option of the ccrh command.		
	Default	%BuildModeName%	
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.	
	Restriction	Up to 247 characters	

(3) [Frequently Used Options(for Compile)]

The detailed information on frequently used options during compilation is displayed and the configuration can be changed.

<table>
<tr><td>Level of optimization</td><td colspan="3">Select the level of the optimization for compiling.
This corresponds to the -O option of the ccrh command.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">Default Optimization(None)</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td></td><td>Restriction</td><td>Default Optimization(None)</td><td>Performs optimization that debugging is not affected (optimization of expressions and register allocation, and the like).</td></tr>
<tr><td></td><td></td><td>Code Size Precedence(-Osize)</td><td>Performs optimization with the object size precedence.
Regards reducing the ROM/RAM capacity as important and performs the maximum optimization that is effective for general programs.</td></tr>
<tr><td></td><td></td><td>Speed Precedence(-Ospeed)</td><td>Performs optimization with the execution speed precedence.
Regards shortening the execution speed as important and performs the maximum optimization that is effective for general programs.</td></tr>
<tr><td></td><td></td><td>Debug Precedence(-Onothing)</td><td>Performs optimization with the debug precedence.
Regards debugging as important and suppresses all optimization including default optimization.</td></tr>
<tr><td>Additional include paths</td><td colspan="3">Specify the additional include paths during compiling.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
The specified include path is searched with higher priority than the standard include file folder of CC-RH.
The reference point of the path is the project folder.
When this property is omitted, only the standard folder of CC-RH is searched.
This corresponds to the -I option of the ccrh command.
The specified include path is displayed as the subproperty.
When the include path is added to the project tree, the path is added to the top of the subproperties.
Uppercase characters and lowercase characters are not distinguished for the include paths.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">Additional include paths[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Edit by the Path Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td colspan="2">Up to 259 characters
Up to 256 items can be specified.</td></tr>
</table>

<table>
<tr><td rowspan="4">System include paths</td><td colspan="2">Change the specified order of the include paths which the system set during compiling.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
The system include path is searched with lower priority than the additional include path.
The reference point of the path is the project folder.
This corresponds to the -I option of the ccrh command.
The include path is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>System include paths[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the System Include Path Order dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td>Changes not allowed (Only the specified order of the include paths can be changed.)</td></tr>
<tr><td rowspan="4">Macro definition</td><td colspan="2">Specify the name of the macro to be defined.
Specify in the format of "macro name=defined value", with one macro name per line.
The "=defined value" part can be omitted, and in this case, "1" is used as the defined value.
This corresponds to the -D option of the ccrh command.
The specified macro is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Macro definition[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 256 characters
Up to 256 items can be specified.</td></tr>
</table>

(4) [Frequently Used Options(for Assemble)]

The detailed information on frequently used options during assembling is displayed and the configuration can be changed.

This category is displayed when [No] in the [Build simultaneously] property in the [Build Method] category is selected.

<table>
<tr><td>Additional include paths</td><td colspan="2">Specify the additional include paths during assembling.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
The specified include path is searched with higher priority than the standard include file folder of CC-RH.
The reference point of the path is the project folder.
When this property is omitted, only the standard folder of CC-RH is searched.
This corresponds to the -I option of the ccrh command.
The specified include path is displayed as the subproperty.
When the include path is added to the project tree, the path is added to the top of the subproperties.
Uppercase characters and lowercase characters are not distinguished for the include paths.</td></tr>
<tr><td></td><td>Default</td><td>Additional include paths[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td>Edit by the Path Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 259 characters
Up to 256 items can be specified.</td></tr>
<tr><td>System include paths</td><td colspan="2">Change the specified order of the include paths which the system set during assembling.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
The system include path is searched with lower priority than the additional include path.
The reference point of the path is the project folder.
This corresponds to the -I option of the ccrh command.
The include path is displayed as the subproperty.</td></tr>
<tr><td></td><td>Default</td><td>System include paths[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td>Edit by the System Include Path Order dialog box which appears when clicking the [...] button.</td></tr>
<tr><td></td><td>Restriction</td><td>Changes not allowed (Only the specified order of the include paths can be changed.)</td></tr>
</table>

Macro definition	Specify the name of the macro to be defined. Specify in the format of "*macro name*=*defined value*", with one macro name per line. The "=*defined value*" part can be omitted, and in this case, "1" is used as the defined value. This corresponds to the -D option of the ccrh command. The specified macro is displayed as the subproperty.	
	Default	Macro definition[*number of defined items*]
	How to change	Edit by the Text Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.
	Restriction	Up to 256 characters Up to 256 items can be specified.

(5) [Frequently Used Options(for Link)]

The detailed information on frequently used options during linking is displayed and the configuration can be changed.

This category is not displayed for the library project.

Using libraries	Specify the library files to be used. The following placeholders are supported. %ActiveProjectDir%: Replaces with the absolute path of the active project folder. %ActiveProjectName%: Replaces with the active project name. %BuildModeName%: Replaces with the build mode name. %MainProjectDir%: Replaces with the absolute path of the main project folder. %MainProjectName%: Replaces with the main project name. %MicomToolPath%: Replaces with the absolute path of the install folder of this product. %ProjectDir%: Replaces with the absolute path of the project folder. %ProjectName%: Replaces with the project name. %TempDir%: Replaces with the absolute path of the temporary folder. %WinDir%: Replaces with the absolute path of the Windows system folder. This corresponds to the -LIBrary option of the rlink command. The library file name is displayed as the subproperty.	
	Default	Using libraries[*number of defined items*]
	How to change	Edit by the Path Edit dialog box which appears when clicking the [...] button. -> Edit by the Specify Using Library File dialog box which appears when clicking the [Browse...] button. For the subproperty, you can enter directly in the text box.
	Restriction	Up to 259 characters Up to 65536 items can be specified.

<table>
<tr><td>Output folder</td><td colspan="3">Specify the output folder.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
If this is blank, it is assumed that the project folder has been specified.
This corresponds to the -OUtput option of the rlink command.
This property is displayed only when [No] in the [Divide load module file] property is selected.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">%BuildModeName%</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.</td></tr>
<tr><td></td><td>Restriction</td><td colspan="2">Up to 247 characters</td></tr>
<tr><td>Output file name</td><td colspan="3">Specify the output file name.
If the extension is omitted, ".abs" is automatically added.
The following placeholders are supported.
%ActiveProjectName%: Replaces with the active project name.
%MainProjectName%: Replaces with the main project name.
%ProjectName%: Replaces with the project name.
This corresponds to the -OUtput option of the rlink command.
This property is displayed only when [No] in the [Divide load module file] property is selected.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">%ProjectName%.abs</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Directly enter in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td colspan="2">Up to 259 characters</td></tr>
<tr><td>Use standard libraries</td><td colspan="3">Select whether to use the standard libraries provided by the compiler.
This corresponds to the -LIBrary option of the rlink command.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td></td><td rowspan="2">Restriction</td><td>Yes</td><td>Uses the standard libraries.</td></tr>
<tr><td></td><td>No</td><td>Does not use the standard libraries.</td></tr>
</table>

Use "Standard Library" function	Select whether to use the standard library functions. This property is changed to [Yes(-LIBrary=libc)] when [Yes] in the [Use "Mathematical Library (Double precision)" function] or [Use "Mathematical Library (Single precision)" function] property is selected. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	Yes(-LIBrary=libc)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libc)	Uses the standard library functions.
		No	Does not use the standard library functions.
Use "Mathematical Library (Double precision)" function	Select whether to use the mathematical library (double precision) functions. This property is changed to [No] when [No] in the [Use standard libraries] property is selected. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	Yes(-LIBrary=libm)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libm)	Uses the mathematical library (double precision) functions.
		No	Does not use the mathematical library (double precision) functions.
Use "Mathematical Library (Single precision)" function	Select whether to use the mathematical library (single precision) functions. This property is changed to [No] when [No] in the [Use standard libraries] property is selected. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	Yes(-LIBrary=libmf)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libmf)	Uses the mathematical library (single precision) functions.
		No	Does not use the mathematical library (single precision) functions.
Use "Non-local jump Library" function	Select whether to use the non-local jump library functions. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libsetjmp)	Uses the non-local jump library functions.
		No	Does not use the non-local jump library functions.

(6) [Frequently Used Options(for Hex Output)]

The detailed information on frequently used options during hex outputting is displayed and the configuration can be changed.

This category is not displayed for the library project.

<table>
<tr><td rowspan="5">Output hex file</td><td colspan="3">Select whether to output the hex file.
This corresponds to the -FOrm option of the rlink command.</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td>Restriction</td><td>Yes</td><td>Outputs the hex file.</td></tr>
<tr><td>Default</td><td>No</td><td>Does not output the hex file.</td></tr>
<tr><td rowspan="6">Hex file format</td><td colspan="3">Select the format of the hex file to be output.
This corresponds to the -FOrm option of the rlink command.
This property is displayed only when [Yes] in the [Output hex file] property is selected.</td></tr>
<tr><td>Default</td><td colspan="2">Motorola S-record file(-FOrm=Stype)</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="3">Restriction</td><td>Intel HEX file(-FOrm=Hexadecimal)</td><td>Outputs an Intel HEX file.</td></tr>
<tr><td>Motorola S-record file(-FOrm=Stype)</td><td>Outputs a Motorola S-record file.</td></tr>
<tr><td>Binary file(-FOrm=Binary)</td><td>Outputs a binary file.</td></tr>
<tr><td rowspan="4">Output folder</td><td colspan="3">Specify the folder which the hex file is output.
If a relative path is specified, the reference point of the path is the main project or subproject folder.
If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
If this is blank, it is assumed that the project folder has been specified.
This corresponds to the -OUtput option of the rlink command.
This property is displayed only when [Yes] in the [Output hex file] property is selected.</td></tr>
<tr><td>Default</td><td colspan="2">%BuildModeName%</td></tr>
<tr><td>How to change</td><td colspan="2">Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td colspan="2">Up to 247 characters</td></tr>
</table>

<table>
<tr><td>Output file name</td><td colspan="2">Specify the hex file name.
If the extension is omitted, it is automatically added according to the selection in the [Hex file format] property.
When [Intel HEX file(-FOrm=Hexadecimal)] is selected: .hex
When [Motorola S-record file(-FOrm=Stype)] is selected: .mot
When [Binary file(-FOrm=Binary)] is selected: .bin
The following placeholders are supported.
%ActiveProjectName%: Replaces with the active project name.
%MainProjectName%: Replaces with the main project name.
%ProjectName%: Replaces with the project name.
This corresponds to the -OUtput option of the rlink command.
This property is displayed only when [Yes] in the [Output hex file] property is selected.</td></tr>
<tr><td></td><td>Default</td><td>%ProjectName%.mot</td></tr>
<tr><td></td><td>How to change</td><td>Directly enter in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 259 characters</td></tr>
<tr><td>Division output file</td><td colspan="2">Specify the division output files.
Specify in the format of "file name=start address-end address" or "file name=section name", with one entry per line.
If multiple section names are specified, delimit them with a colon as in "file name=section name:section name" (example: file1.mot=sec1:sec2).
Specify the address in hexadecimal (example: file2.mot=400-4ff).
If the extension is omitted, it is automatically added according to the selection in the [Hex file format] property.
When [Intel HEX file(-FOrm=Hexadecimal)] is selected: .hex
When [Motorola S-record file(-FOrm=Stype)] is selected: .mot
When [Binary file(-FOrm=Binary)] is selected: .bin
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
This corresponds to the -OUtput option of the rlink command.
The division output file name is displayed as the subproperty.
This property is displayed only when [Yes] in the [Output hex file] property is selected.</td></tr>
<tr><td></td><td>Default</td><td>Division output file[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 259 characters
Up to 65535 items can be specified.</td></tr>
</table>

(7) [Frequently Used Options(for Create Library)]

The detailed information on frequently used options during library generation is displayed and the configuration can be changed.

This category is displayed only for the library project.

Output file format	Select the format of the output file. This corresponds to the -FOrm option of the rlink command.		
	Default	User libraries(-FOrm=Library=U)	
	How to change	Select from the drop-down list.	
	Restriction	User libraries(-FOrm=Library=U)	Outputs a user library file.
		System libraries(-FOrm=Library=S)	Outputs a system library file.
		Relocatable file(-FOrm=Relocate)	Outputs a relocatable file.
Output folder	Specify the output folder. The following placeholders are supported. %ActiveProjectDir%: Replaces with the absolute path of the active project folder. %ActiveProjectName%: Replaces with the active project name. %BuildModeName%: Replaces with the build mode name. %MainProjectDir%: Replaces with the absolute path of the main project folder. %MainProjectName%: Replaces with the main project name. %MicomToolPath%: Replaces with the absolute path of the install folder of this product. %ProjectDir%: Replaces with the absolute path of the project folder. %ProjectName%: Replaces with the project name. %TempDir%: Replaces with the absolute path of the temporary folder. %WinDir%: Replaces with the absolute path of the Windows system folder. If this is blank, it is assumed that the project folder has been specified. This corresponds to the -OUtput option of the rlink command.		
	Default	%BuildModeName%	
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.	
	Restriction	Up to 247 characters	
Output file name	Specify the output file name. If the extension is omitted, it is automatically added according to the selection in the [Hex file format] property. When [User libraries(-FOrm=Library=U)] is selected: .lib When [System libraries(-FOrm=Library=S)] is selected: .lib When [Relocatable file(-FOrm=Relocate)] is selected: .rel The following placeholders are supported. %ActiveProjectName%: Replaces with the active project name. %MainProjectName%: Replaces with the main project name. %ProjectName%: Replaces with the project name. This corresponds to the -OUtput option of the rlink command.		
	Default	%ProjectName%.lib	
	How to change	Directly enter in the text box.	
	Restriction	Up to 259 characters	

Use standard libraries	Select whether to use the standard libraries provided by the compiler. This corresponds to the -LIBrary option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes	Uses the standard libraries.
		No	Does not use the standard libraries.
Use "Standard Library" function	Select whether to use the standard library functions. This property is changed to [Yes(-LIBrary=libc)] when [Yes] in the [Use "Mathematical Library (Double precision)" function] or [Use "Mathematical Library (Single precision)" function] property is selected. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libc)	Uses the standard library functions.
		No	Does not use the standard library functions.
Use "Mathematical Library (Double precision)" function	Select whether to use the mathematical library (double precision) functions. This property is changed to [No] when [No] in the [Use standard libraries] property is selected. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libm)	Uses the mathematical library (double precision) functions.
		No	Does not use the mathematical library (double precision) functions.
Use "Mathematical Library (Single precision)" function	Select whether to use the mathematical library (single precision) functions. This property is changed to [No] when [No] in the [Use standard libraries] property is selected. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libmf)	Uses the mathematical library (single precision) functions.
		No	Does not use the mathematical library (single precision) functions.

Use "Non-local jump Library" function	Select whether to use the non-local jump library functions. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libsetjmp)	Uses the non-local jump library functions.
		No	Does not use the non-local jump library functions.

(8) [Register Mode]

The detailed information on register modes is displayed and the configuration can be changed.

Register mode	Select the register mode (number of registers used by the Compiler)[Note] of the software register bank function. This corresponds to the -Xreg_mode option of the ccrh command.		
	Default	32-register mode(None)	
	How to change	Select from the drop-down list.	
	Restriction	32-register mode(None)	Sets the register mode to 32.
		22-register mode(-Xreg_mode=22)	Sets the register mode to 22.
		Universal register mode(-Xreg_mode=common)	Sets the register mode to 22. Use this item to generate the object module file that does not depend on the register mode.
Reserve r2 register	Select whether to reserve the r2 register. This corresponds to the -Xreserve_r2 option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xreserve_r2)	Reserves the r2 register and generates code that does not use this register by the compiler.
		No	The compiler uses the r2 register without reserving it.
ep-register treatment	Select how to handle the ep register. Specify [Fix in project(-Xep=fix)] when any of the following attribute strings in a #pragma section in the project is used. ep_auto, ep_disp4, ep_disp5, ep_disp7, ep_disp8, ep_disp16, ep_disp23 Only [Treat as callee-save(None)] is displayed when [Yes] in the [Optimize accesses to external variables] property in the [Optimization(Details)] category from the [Compile Options] tab is selected. This corresponds to the -Xep option of the ccrh command.		
	Default	Treat as callee-save(None)	
	How to change	Select from the drop-down list.	
	Restriction	Treat as callee-save(None)	Fixes the value of the ep register for the entire project.
		Fix in project(-Xep=fix)	Treats the ep register as a register guaranteeing the value before and after the function call.

Note The register modes provided by CC-RH are shown below.

Register Mode	Working Registers	Registers for Register Variables
common	r10 to r14	r25 to r29
22-register mode	r10 to r14	r25 to r29
32-register mode	r10 to r19	r20 to r29

(9) [Error Output]

The detailed information on the error output is displayed and the configuration can be changed.

Output error message file	Select whether to output the error message file. This corresponds to the -Xerror_file option of the ccrh command. Error messages are displayed on the Output panel regardless of this property's .		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xerror_file)	Outputs the error message file.
		No	Does not output the error message file.
Error message file output folder	Specify the folder which the error message file is output. If a relative path is specified, the reference point of the path is the main project or subproject folder. If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different). The following placeholder is supported. %BuildModeName%: Replaces with the build mode name. If this is blank, it is assumed that the project folder has been specified. This corresponds to the -Xerror_file option of the ccrh command. This property is displayed only when [Yes(-Xerror_file)] in the [Output error message file] property is selected.		
	Default	%BuildModeName%	
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.	
	Restriction	Up to 247 characters	
Error message file name	Specify the error message file name. The extension can be freely specified. The following placeholders are supported. %ActiveProjectName%: Replaces with the active project name. %MainProjectName%: Replaces with the main project name. %ProjectName%: Replaces with the project name. If this is blank, it is assumed that "%ProjectName%.err" has been specified. This corresponds to the -Xerror_file option of the ccrh command. This property is displayed only when [Yes(-Xerror_file)] in the [Output error message file] property is selected.		
	Default	%ProjectName%.err	
	How to change	Directly enter in the text box.	
	Restriction	Up to 259 characters	

(10) [Warning Message]

The detailed information on warning messages is displayed and the configuration can be changed.

Undisplayed warning message	Specify the number of the warning message not to be displayed. If multiple message numbers are specified, delimit them with "," (comma) (example: 02042,02107). Also, the range can be set using "-" (hyphen) (example: 02222-02554,02699-02782). This corresponds to the -Xno_warning option of the ccrh command.	
	Default	Blank
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.
	Restriction	Up to 2048 characters

(11) [Build Method]

The detailed information on the build method is displayed and the configuration can be changed.

Build simultaneously	Select whether to geneate the load module file by compiling/assembling/linking multiple files simultaneously. The files with the individual build options and files to be executed prior to the build are excluded from runing a build simultaneously. See "2.12.5 Compile/assemble/link multiple files simultaneously" for details about runing a build simultaneously.		
	Default	Yes	
	How to change	Select from the drop-down list.	
	Restriction	Yes	Compiles, assembles, and links multiple files simultaneously. The assembly source file (except the file with the individual options) is assembled using the settings of the [Compile Options] tab.
		No	Compiles, assembles, and links for each file.
Handling the source file includes non-existing file	Selects whether to recompile/assemble the source file if there are no files that include it.		
	Default	Re-compile/assemble the source file	
	How to change	Select from the drop-down list.	
	Restriction	Re-compile/assemble the source file	Recompiles/assembles the source file if there are no files that include it.
		Ignore re-compiling/ assembling the source file	Does not recompile/assemble the source file even if there are no files that include it.

(12) [Version Select]

The detailed information on the build tool version is displayed and the configuration can be changed.

Using compiler package install folder	The folder in which the compiler package to be used is installed is displayed.	
	Default	*Install folder name*
	How to change	Changes not allowed

Using compiler package version	Select the version of the compiler package to be used. This setting is common to all the build modes.		
	Default	Always latest version which was installed	
	How to change	Select from the drop-down list.	
	Restriction	Always latest version which was installed	Uses the latest version in the installed compiler packages.
		Versions of the installed compiler packages	Uses the selected version in the compiler package.
Latest compiler package version which was installed	The version of the compiler package to be used when [Always latest version which was installed] is selected in the [Using compiler package version] property is displayed. This setting is common to all the build modes. This property is displayed only when [Always latest version which was installed] in the [Using compiler package version] property is selected.		
	Default	*Latest version of the installed compiler packages*	
	How to change	Changes not allowed	

(13) [Notes]

The detailed information on notes is displayed and the configuration can be changed.

Memo	Add memos to the build tool. Add one item in one line. This setting is common to all the build modes. The specified memo is displayed as the subproperty.	
	Default	Memo[*number-of-items*]
	How to change	Edit by the Text Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.
	Restriction	Up to 256 characters Up to 256 items can be specified.

(14) [Others]

Other detailed information on the build tool is displayed and the configuration can be changed.

<table>
<tr><td rowspan="6">Output message format</td><td colspan="3">Specify the format of the message under build execution.
This applies to the messages output by the build tool to be used, and commands added by plugins.
It does not apply to the output messages of commands specified in the [Commands executed before build processing] or [Commands executed after build processing] property.
The following placeholders are supported.
%Options%: Replaces with the command line option under build execution.
%Program%: Replaces with the program name under execution.
%TargetFiles%: Replaces with the file name being compile/assemble or making link.
If this is blank, "%Program% %Options%" will be set automatically.</td></tr>
<tr><td>Default</td><td colspan="2">%TargetFiles%</td></tr>
<tr><td>How to change</td><td colspan="2">Directly enter in the text box (up to 256 characters) or select from the drop-down list.</td></tr>
<tr><td rowspan="3">Restriction</td><td>%TargetFiles%</td><td>Displays the file name in the output message.</td></tr>
<tr><td>%TargetFiles%: %Options%</td><td>Displays the file name and command line options in the output message.</td></tr>
<tr><td>%Program% %Options%</td><td>Displays the program name and command line options in the output message.</td></tr>
<tr><td rowspan="4">Format of build option list</td><td colspan="3">Specify the display format of the build option list (see "2.11.4 Display a list of build options").
This applies to the options of the build tool to be used, and commands added by plugins.
It does not apply to the options of commands specified in the [Commands executed before build processing] or [Commands executed after build processing] property.
The following placeholders are supported.
%Options%: Replaces with the command line option under build execution.
%Program%: Replaces with the program name under execution.
%TargetFiles%: Replaces with the file name being compile/assemble or making link.
If this is blank, "%TargetFiles% : %Program% %Options%" will be set automatically.</td></tr>
<tr><td>Default</td><td colspan="2">%TargetFiles% : %Program% %Options%</td></tr>
<tr><td>How to change</td><td colspan="2">Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td colspan="2">Up to 256 characters</td></tr>
</table>

<table>
<tr><td rowspan="4">Commands executed before build processing</td><td colspan="2">Specify the command to be executed before build processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed before build processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Commands executed before build processing[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr>
</table>

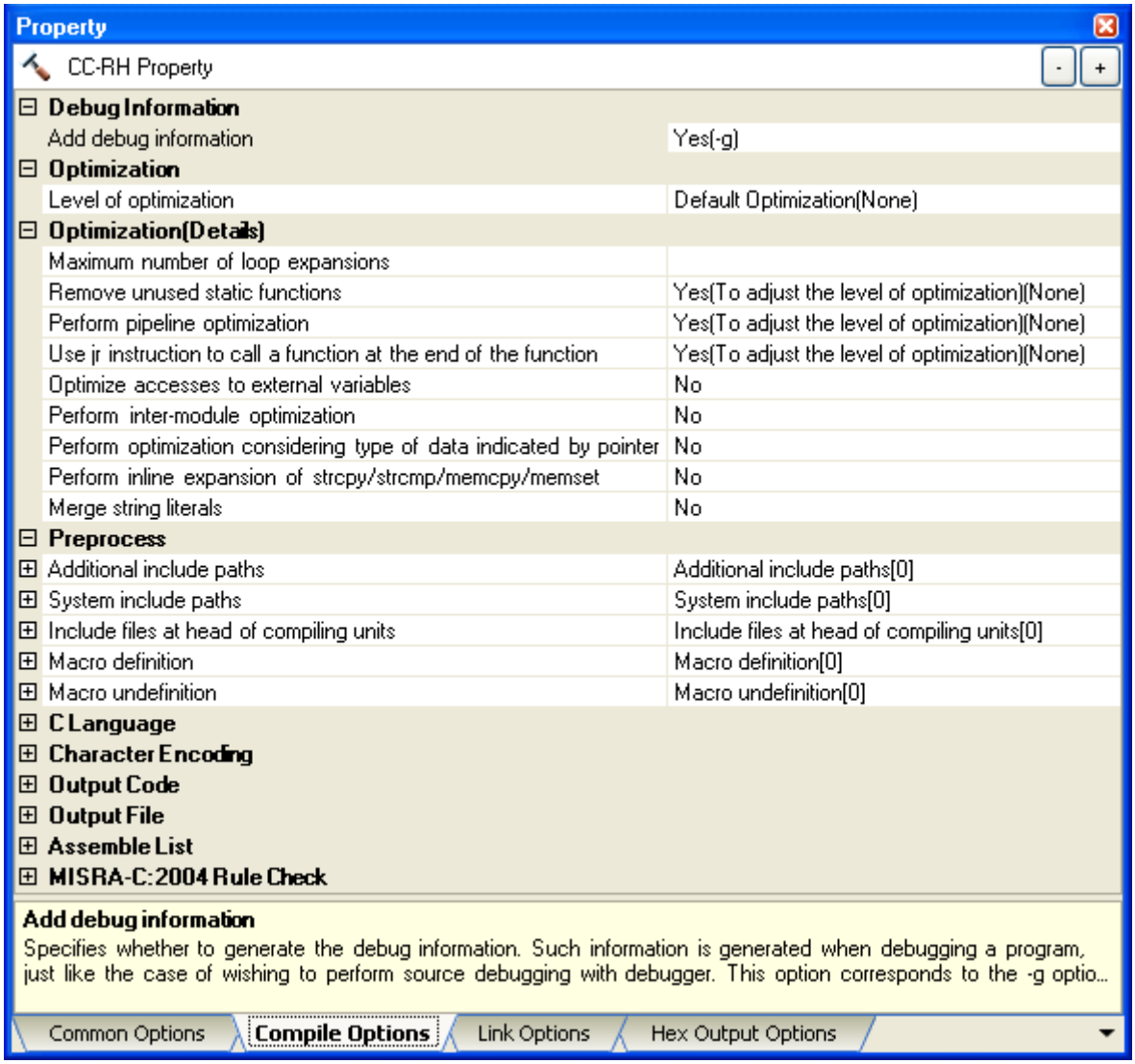
<table>
<tr><td rowspan="4">Commands executed after build processing</td><td colspan="2">Specify the command to be executed after build processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed after build processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Commands executed after build processing[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr>
<tr><td rowspan="4">Other additional options</td><td colspan="2">Input the option to be added additionally.
The options set here are added at the end of the ccrh options group.</td></tr>
<tr><td>Default</td><td>Blank</td></tr>
<tr><td>How to change</td><td>Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td>Up to 259 characters</td></tr>
</table>

[Compile Options] tab

This tab shows the detailed information on the compile phase categorized by the following and the configuration can be changed.

(1) [Debug Information]
(2) [Optimization]
(3) [Optimization(Details)]
(4) [Preprocess]
(5) [C Language]
(6) [Character Encoding]
(7) [Output Code]
(8) [Output File]
(9) [Assemble List]
(10) [MISRA-C:2004 Rule Check]
(11) [Others]

Figure A-5. Property Panel: [Compile Options] Tab



[Description of each category]

(1) [Debug Information]

The detailed information on debug information is displayed and the configuration can be changed.

Add debug information	Select whether to generate the debug information. It is possible to perform source debugging with the debugger by outputting information for source debugging to the output file. This corresponds to the -g option of the ccrh command.		
	Default	Yes(-g)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-g)	Generates the debug information.
		No	Does not generate the debug information.

(2) [Optimization]

The detailed information on the optimization is displayed and the configuration can be changed.

Level of optimization	Select the level of the optimization for compiling. This corresponds to the -O option of the ccrh command.		
	Default	Default Optimization(None)	
	How to change	Select from the drop-down list.	
	Restriction	Default Optimization(None)	Performs optimization that debugging is not affected (optimization of expressions and register allocation, and the like).
		Code Size Precedence(-Osize)	Performs optimization with the object size precedence. Regards reducing the ROM/RAM capacity as important and performs the maximum optimization that is effective for general programs.
		Speed Precedence(-Ospeed)	Performs optimization with the execution speed precedence. Regards shortening the execution speed as important and performs the maximum optimization that is effective for general programs.
		Debug Precedence(-Onothing)	Performs optimization with the debug precedence. Regards debugging as important and suppresses all optimization including default optimization.

(3) [Optimization(Details)]

The detailed information on the optimization is displayed and the configuration can be changed.

Maximum number of loop expansions	Specify the maximum number of times to expand the loops such as "for" and "while". If 0 or 1 is specified, expansion is suppressed. If this is blank, it is assumed that "4" has been specified. This corresponds to the -Ounroll option of the ccrh command.		
	Default	Blank	
	How to change	Directly enter in the text box.	
	Restriction	0 to 999 (decimal number) or blank	
Remove unused static functions	Select whether to remove the static functions which are not called. This corresponds to the -Odelete_static_func option of the ccrh command.		
	Default	Yes(To adjust the level of optimization)(None)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(To adjust the level of optimization)(None)	Performs optimization according to the [Level of optimization] property.
		Yes(-Odelete_static_func)	Removes the unused static functions which are not called.
		No(-Odelete_static_func=off)	Does not remove the unused static functions which are not called.
Perform inline expansion	Specify whether to perform inline expansion at the location calling functions. This corresponds to the -Oinline option of the ccrh command. This property is displayed only when [Code Size Precedence(-Osize)] or [Speed Precedence(-Ospeed)] in the [Level of optimization] property is selected.		
	Default	Yes(To adjust the level of optimization)(None)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(To adjust the level of optimization)(None)	Performs optimization according to the [Level of optimization] property.
		Yes(Only specified functions)(-Oinline=1)	Performs inline expansion at the location calling the function for which #pragma inline is specified.
		Yes(Auto-detect)(-Oinline=2)	Distinguishes the function that is the target of inline expansion automatically and expands it.
		Yes(Auto-detect without code size increase)(-Oinline=3)	Distinguishes the function that is the target of inline expansion automatically and expands it, while minimizing the increase in code size.
		No(-Oinline=0)	Suppresses all inline expansion including the function for which "#pragma inline" is specified.

Maximum increasing rate of inline expansion size	Specify the maximum increasing rate (%) of the code size up to which inline expansion is performed. (Example: When "100" is specified, inline expansion will be applied until the code size increases by 100% (becomes twice the initial size).) This corresponds to the -Oinline_size option of the ccrh command. This property is displayed only when [Yes(Auto-detect)(-Oinline=2)] in the [Perform inline expansion] property is selected, or when [Yes(To adjust the level of optimization)] in the [Perform inline expansion] property and [Speed Precedence(-Ospeed)] in the [Optimization Level] property are selected.		
	Default	100	
	How to change	Directly enter in the text box.	
	Restriction	0 to 65535 (decimal number)	
Perform pipeline optimization	Select whether to improve the program's execution performance by reordering instructions at the machine-language level. This corresponds to the -Opipeline option of the ccrh command.		
	Default	Yes(To adjust the level of optimization)(None)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(To adjust the level of optimization)(None)	Performs optimization according to the [Level of optimization] property.
		Yes(-Opipeline)	Performs pipeline optimization.
		No(-Opipeline=off)	Does not perform pipeline optimization.
Use jr instruction to call a function at the end of the function	Select whether to give precedence to using jr instructions in the place of jarl instructions when the function ends with a function call. This corresponds to the -Otail_call option of the ccrh command.		
	Default	Yes(To adjust the level of optimization)(None)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(To adjust the level of optimization)(None)	Performs optimization according to the [Level of optimization] property.
		Yes(-Otail_call)	Gives precedence to using jr instructions in the place of jarl instructions when the function ends with a function call. The code size can be reduced by removing the store/restore instructions for lp. However, some debug functions cannot be used.
		No(-Otail_call=off)	Uses jarl instructions when the function ends with a function call.

Optimize accesses to external variables	Select whether to optimize accesses to external variables. This corresponds to the -Osmap and -Omap options of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(Optimizes the inner-module)(-Osmap)	Sets a base address for external or static variables defined in the file to be compiled, and generates code that accesses these relative to the base address.
		Yes(Optimizes the inter-module)(-Omap)	Generates an external symbol allocation information file. According to the information, recompilation is done to generate code that performs access to external or static variables relative to the base address. This item is displayed only when [No] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected. If [Yes] in the [Build simultaneously] property is selected when this item is selected, this item will be changed to [No].
		No	Does not optimize accesses to external variables.
Perform inter-module optimization	Specify the level of inter-module optimization (such as function merging). Only [Yes(Level 1)(Perform)(-Xintermodule)] and [No] are displayed when [No] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected. This corresponds to the -Xwhole_program, -Xmerge_files, and -Xintermodule options of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(Level 3)(Perform with assuming it the whole program)(-Xwhole_program)	Performs inter-module optimization assuming that the source files comprise the entire program. However, operation is not guaranteed if the preconditions are not met. See "-Xwhole_program" for details about the preconditions.
		Yes(Level 2)(Perform with merging files)(-Xmerge_files, -Xintermodule)	Merges two or more C source files and performs inter-module optimization. This item is displayed only when two or more source files are added to the project.
		Yes(Level 1)(Perform)(-Xintermodule)	Performs inter-module optimization for each file.
		No	Does not perform inter-module optimization.

Perform optimization considering type of data indicated by pointer	Select whether to perform optimization with consideration for the type of the data indicated by the pointer, based on the ANSI standard. This corresponds to the -Xalias option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xalias=ansi)	Performs optimization with consideration for the type of the data indicated by the pointer. In general, this option improves the object performance, but the execution result may differ from the case when [No] is selected.
		No	Does not perform optimization with consideration for the type of the data indicated by the pointer.
Perform inline expansion of strcpy/strcmp/ memcpy/memset	Selsect whether to perform inline expansion of functions "strcpy()", "strcmp()", "memcpy()", and "memset()" calls, with regarding the alignment conditions of the array (including character strings) and the structure as 4 bytes. This improves the execution speed of the program to be generated, but it increases the code size. This corresponds to the -Xinline_strcpy option of the ccrh command. This property is displayed only when [No] in the [Structure packing] property in the [Output Code] category is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xinline_strcpy)	Performs inline expansion of functions "strcpy()", "strcmp()", "memcpy()", and "memset()" calls.
		No	Does not perform inline expansion of functions "strcpy()", "strcmp()", "memcpy()", and "memset()" calls.
Merge string literals	When the same string literals exist in the source file, specify whether to merge them and allocate to the one area. This corresponds to the -Xmerge_string option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xmerge_string)	Merges the same string literals exist in the source file and allocates to the one area.
		No	Each allocates the same string literals exist in the source file to separate areas.

(4) [Preprocess]

The detailed information on preprocessing is displayed and the configuration can be changed.

<table>
<tr><td>Additional include paths</td><td colspan="2">Specify the additional include paths during compiling.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
The specified include path is searched with higher priority than the standard include file folder of CC-RH.
The reference point of the path is the project folder.
When this property is omitted, only the standard folder of CC-RH is searched.
This corresponds to the -I option of the ccrh command.
The specified include path is displayed as the subproperty.
When the include path is added to the project tree, the path is added to the top of the subproperties.
Uppercase characters and lowercase characters are not distinguished for the include paths.</td></tr>
<tr><td></td><td>Default</td><td>Additional include paths[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td>Edit by the Path Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 259 characters
Up to 256 items can be specified.</td></tr>
<tr><td>System include paths</td><td colspan="2">Change the specified order of the include paths which the system set during compiling.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
The system include path is searched with lower priority than the additional include path.
The reference point of the path is the project folder.
This corresponds to the -I option of the ccrh command.
The include path is displayed as the subproperty.</td></tr>
<tr><td></td><td>Default</td><td>System include paths[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td>Edit by the System Include Path Order dialog box which appears when clicking the [...] button.</td></tr>
<tr><td></td><td>Restriction</td><td>Changes not allowed (Only the specified order of the include paths can be changed.)</td></tr>
</table>

<table>
<tr><td rowspan="4">Include files at head of compiling units</td><td colspan="2">Specify the file that is included at the top of the compilation unit.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
The reference point of the path is the project folder.
This corresponds to the -Xpreinclude option of the ccrh command.
The specified include file name is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Include files at head of compiling units[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 247 characters
Up to 256 items can be specified.</td></tr>
<tr><td rowspan="4">Macro definition</td><td colspan="2">Specify the name of the macro to be defined.
Specify in the format of "macro name=defined value", with one macro name per line.
The "=defined value" part can be omitted, and in this case, "1" is used as the defined value.
This corresponds to the -D option of the ccrh command.
The specified macro is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Macro definition[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 256 characters
Up to 256 items can be specified.</td></tr>
<tr><td rowspan="4">Macro undefinition</td><td colspan="2">Specify the macro name to be undefined.
Specify in the format of "macro name", with one macro name per line.
This corresponds to the -U option of the ccrh command.
The specified macro is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Macro undefinition[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 256 characters
Up to 256 items can be specified.</td></tr>
</table>

Output C source comments to preprocessed file	Select whether to output the comments of the C source to the preprocessed file. This corresponds to the -Xpreprocess option of the ccrh command. This property is displayed only when [Yes(-P)] in the [Output preprocessed source file] property in the [Output File] category is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xpreprocess=comment)	Outputs the comments of the C source to the preprocessed file.
		No	Does not output the comments of the C source to the preprocessed file.
Output line number information to preprocessed file	Select whether to output the line number information of the C source to the preprocessed file. This corresponds to the -Xpreprocess option of the ccrh command. This property is displayed only when [Yes(-P)] in the [Output preprocessed source file] property in the [Output File] category is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xpreprocess=line)	Outputs the line number information of the C source to the preprocessed file.
		No	Does not output the line number information of the C source to the preprocessed file.

(5) [C Language]

The detailed information on C language is displayed and the configuration can be changed.

Compile strictly according to ANSI standards	Select whether to process as making C source program comply strictly with the ANSI standard and output an error or warning for a specification that violates the standard. This corresponds to the -Xansi option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xansi)	Processes as making C source program comply strictly with the ANSI standard and outputs an error or warning for a specification that violates the standard.
		No	Compatibility with the conventional C language specifications is conferred and processing continues after warning is output.
Enumeration type	Select which integer type the enumeration type handles. This corresponds to the -Xenum_type option of the ccrh command.		
	Default	signed int(None)	
	How to change	Select from the drop-down list.	
	Restriction	signed int(None)	The enumeration type is handled as int type.
		auto(-Xenum_type=auto)	Handles each enumerated type as the smallest integer type capable of expressing all the enumerators in that type.

Handle external variables as if they are volatile qualified	Select whether to handle all external variables as if they were volatile-declared. This corresponds to the -Xvolatile option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xvolatile)	Handles all external variables as if they were volatile-declared.
		No	Handles only the volatile-qualified variables as they were volatile-declared.
Check C program compatibility	Select whether to check the compatibility of a C program. This corresponds to the -Xcheck option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(for SuperH RISC engine C/C++ compiler)(-Xcheck=shc)	Checks the compatibility with the SuperH family C/C++ compiler.
		No	Does not check the compatibility with existing programs.

(6) [Character Encoding]

The detailed information on character encoding is displayed and the configuration can be changed.

Character encoding	Select the character code to be used for Japanese/Chinese comments and character strings in the source file. This corresponds to the -Xcharacter_set option of the ccrh command.		
	Default	Auto(None)	
	How to change	Select from the drop-down list.	
	Restriction	Auto(None)	Interprets the Japanese character code in the source file as SJIS.
		SJIS(-Xcharacter_set=sjis)	Interprets the Japanese character code in the source file as SJIS.
		EUC(-Xcharacter_set=euc_jp)	Interprets the Japanese character code in the source file as EUC.
		UFT-8(-Xcharacter_set=utf8)	Interprets the Japanese character code in the source file as UFT-8.
		Big5(-Xcharacter_set=big5)	Interprets the Chinese character code in the source file as Traditional Chinese.
		GB2312(-Xcharacter_set=gb2312)	Interprets the Chinese character code in the source file as Simplified Chinese.
		No-process(-Xcharacter_set=none)	Does not interpret the Japanese/Chinese character code in the source file.

(7) [Output Code]

The detailed information on output codes is displayed and the configuration can be changed.

Structure packing	Select whether to perform structure packing. The specified alignment can be used without aligning structure members according to the type of each member. This corresponds to the -Xpack option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	1 byte(-Xpack=1)	Aligns structure members on a 1-byte boundary.
		2 bytes(-Xpack=2)	Aligns structure members on a 2-byte boundary.
		4 bytes(-Xpack=4)	Aligns structure members on a 4-byte boundary.
		No	Does not perform structure packing.
Order of bit-field members	Select the order of bit-field members. This corresponds to the -Xbit_order option of the ccrh command.		
	Default	Allocates from right(None)	
	How to change	Select from the drop-down list.	
	Restriction	Allocates from left(-Xbit_order=left)	Allocates the members from the upper bit.
		Allocates from right(None)	Allocates the members from the lower bit.
Output comment to assembly source file	Select whether to output a C source program as a comment to the assembly source file to be output. This corresponds to the -Xpass_source option of the ccrh command. This property is displayed only when [Yes(-Xasm_path)] in the [Output assembly source file] property in the [Output File] category is selected or when [Yes(-Xasm_option=-Xprn_path)] in the [Output assemble list file] property in the [Assemble List] category is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xpass_source)	Outputs a C source program as a comment to the assembly source file.
		No	Does not output a C source program as a comment to the assembly source file.

Output code of switch statement	Select the code output mode for switch statements in programs. This corresponds to the -Xswitch option of the ccrh command.		
	Default	Auto(None)	
	How to change	Select from the drop-down list.	
	Restriction	Auto(None)	The ccrh selects the optimum output format.
		if-else(-Xswitch=ifelse)	Outputs the switch statements in the same format as the if-else statement along a string of case statements in programs. Select this item if the case statements are written in the order of frequency or if only a few labels are used. Because the case statements are compared starting from the top, unnecessary comparison can be reduced and the execution speed can be increased if the case statement that most often matches is written first.
		Binary search(-Xswitch=binary)	Outputs the code in the binary search format for switch statements in programs. Searches for a matching case statement by using a binary search algorithm. If this item is selected when many labels are used, any case statement can be found at almost the same speed.
		Table jump(-Xswitch=table)	Outputs the code in the table jump format for switch statements in programs. References a table indexed on the values in the case statements, and selects and processes case labels from the switch statement values. The code will branch to all the case statements with about the same speed. However, if case values are not used in succession, an unnecessary area will be created.
Floating-point calculating type	Select whether to generate runtime library call instructions for floating-point calculations, or to generate floating-point instructions for the floating point unit (FPU). This corresponds to the -Xfloat option of the ccrh command. This property is displayed only when [Object for G3M(-Xcpu=g3m)] in the [Specify CPU core] property in the [Output File Type and Path] category from the [Common Options] tab is selected.		
	Default	Auto(None)	
	How to change	Select from the drop-down list.	
	Restriction	Auto(None)	Generates floating-point calculation instructions.
		Software Calculating(-Xfloat=soft)	Generates runtime library call instructions for floating-point calculations.
		FPU Calculating(-Xfloat=fpu)	Generates floating-point calculation instructions of FPU for floating-point calculations.

<table>
<tr><td rowspan="5">Generate div/divu instructions</td><td colspan="3">Select whether to generate the div and divu instructions instead of the divq and divqu instructions for division.
Although the divq and divqu instructions are fast, the number of execution cycles will differ depending on the values of the operands.
This corresponds to the -Xdiv option of the ccrh command.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Restriction</td><td>Yes(-Xdiv)</td><td>Generates the div and divu instructions for division.</td></tr>
<tr><td>No</td><td>Generates the divq and divqu instructions for division.</td></tr>
<tr><td rowspan="5">Generate OV flag check code in division operation</td><td colspan="3">Select whether to generate code (fetrap instruction) that checks the OV flag after division instructions and generate an FE level software exception when the OV flag is 1.
This corresponds to the -Xcheck_div_ov option of the ccrh command.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Restriction</td><td>Yes(-Xcheck_div_ov)</td><td>Generates code that checks the OV flag at division.</td></tr>
<tr><td>No</td><td>Generates code that does not check the OV flag at division.</td></tr>
<tr><td rowspan="4">Vector number of fetrap instruction in divide exception</td><td colspan="3">Specify the vector number of the fetrap instruction generated when the OV flag is 1.
This corresponds to the -Xcheck_div_ov option of the ccrh command.
This property is displayed only when [Yes(-Xcheck_div_ov)] in the [Generate OV flag check code in division operation] property is selected.</td></tr>
<tr><td>Default</td><td colspan="2">1</td></tr>
<tr><td>How to change</td><td colspan="2">Directly enter in the text box.</td></tr>
<tr><td>Restriction</td><td colspan="2">1 to 15 (decimal number)</td></tr>
<tr><td rowspan="5">Generate product-sum operation instruction</td><td colspan="3">Select whether to generate product-sum operation instructions (fmaf.s, fmsf.s, fnmaf.s, and fnmsf.s) for single-precision floating-point product-sum operations.
This corresponds to the -Xuse_fmaf option of the ccrh command.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Restriction</td><td>Yes(-Xuse_fmaf)</td><td>Generates product-sum operation instructions for single-precision floating-point product-sum operations.
Specifying this option will accelerate the execution speed but change the operation precision.</td></tr>
<tr><td>No</td><td>Does not generate product-sum operation instructions.</td></tr>
</table>

Check invalid exception in cmpf instruction	Select whether to generate code by using the comparison condition for generating an invalid operation exception when any of the comparison values is a not-a-number in floating-point comparison. This corresponds to the -Xunordered_cmpf option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xunordered_cmpf)	Generates code by using the comparison condition for generating an invalid operation exception when any of the comparison values is a not-a-number in floating-point comparison.
		No	Does not detect invalid operation exceptions in floating-point comparison.
Use 32-bit branch instruction	Select whether to use the far jump function for the jarl and jr instructions. By using the far jump function, it is assumed that the jarl and jr instructions are jarl32 and jr32 instructions, and assembling is performed. This corresponds to the -Xasm_option=-Xasm_far_jump option of the ccrh command. This property is displayed only when [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xasm_option=-Xasm_far_jump)	Assumes that the jarl and jr instructions are jarl32 and jr32 instructions, and performs assembling.
		No	Performs assembly as a jarl or jr instruction.
Specify jump instruction	Specify the instruction to be generated for function-call branches. This corresponds to the -Xcall_jump option of the ccrh command.		
	Default	Create jarl and jr instructions(None)	
	How to change	Select from the drop-down list.	
	Restriction	Create jarl32 and jr32 instructions(-Xcall_jump=32)	Generates the jarl32 and jr32 instructions for function-call branches.
		Create jarl and jr instructions(None)	Generates the jarl and jr instructions for function-call branches.
Far Jump file names	Specify the Far Jump file name. The code that uses the jarl32 and jr32 instruction for branch instructions of functions described in a file is output to the Far Jump file. The ccrh command outputs an error if the function is in a range that cannot be branched to by the jarl or jr directive (±2MB or more), in which case the Far Jump file is used to recompile. Use the extension ".fjp". This corresponds to the -Xfar_jump option of the ccrh command.		
	Default	Far Jump file name	
	How to change	Directly enter in the text box or edit by the Specify Far Jump File dialog box which appears when clicking the [...] button.	
	Restriction	Up to 259 characters	

Type of a generating program	Select the type of the program to be generated. This corresponds to the -Xmulti_level option of the ccrh command. This property is displayed only for the multi-core project.		
	Default	Generate a program for multi-core(-Xmulti_level=1)	
	How to change	Select from the drop-down list.	
	Restriction	Generate a program for single-core(None)	Generates a single-core program. The #pragma pmodule directives in the program are ignored.
		Generate a program for multi-core(-Xmulti_level=1)	Generates a multi-core program. The #pragma pmodule directives in the program become valid and the PM number is added to the end of the section name.

(8) [Output File]

The detailed information on output files is displayed and the configuration can be changed.

Output assembly source file	Select whether to output the assembly source file of the compile result for the C source. This corresponds to the -Xasm_path option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xasm_path)	Outputs the assembly source file of the compile result for the C source.
		No	Does not output the assembly source file of the compile result for the C source.
Output folder for assembly source file	Specify the folder which the assembly source file is output. If a relative path is specified, the reference point of the path is the main project or subproject folder. If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different). The following placeholder is supported. %BuildModeName%: Replaces with the build mode name. The assembly source file is saved under the C source file name with the extension replaced by ".asm". If this is blank, it is assumed that the project folder has been specified. This corresponds to the -Xasm_path option of the ccrh command. This property is displayed only when [Yes(-Xasm_path)] in the [Output assembly source file] property is selected.		
	Default	%BuildModeName%	
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.	
	Restriction	Up to 247 characters	

Output preprocessed source file	Select whether to output the execution result of preprocessing for the source file to a file. This corresponds to the -P option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-P)	Outputs the execution result of preprocessing for the source file to a file.
		No	Does not output the execution result of preprocessing for the source file to a file.
Output folder for preprocessed source file	Specify the folder which the preprocessed source file is output. The file is output under the source file name with the extension replaced by ".i". If a relative path is specified, the reference point of the path is the main project or subproject folder. If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different). The following placeholder is supported. %BuildModeName%: Replaces with the build mode name. If this is blank, it is assumed that the project folder has been specified. This corresponds to the -Xprep_path option of the ccrh command. This property is displayed only when [Yes(-P)] in the [Output preprocessed source file] property is selected.		
	Default	%BuildModeName%	
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.	
	Restriction	Up to 247 characters	

(9) [Assemble List]

The detailed information on the assemble list is displayed and the configuration can be changed.

Output assemble list file	Select whether to output the assemble list file. This corresponds to the -Xasm_option=-Xprn_path option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xasm_option=-Xprn_path)	Outputs the assemble list file.
		No	Does not output the assemble list file.

Output folder for assemble list file	Specify the folder which the assemble list file is output. The assemble list file is output under the source file name with the extension replaced by ".prn". If a relative path is specified, the reference point of the path is the main project or subproject folder. If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different). The following placeholder is supported. %BuildModeName%: Replaces with the build mode name. If this is blank, it is assumed that the project folder has been specified. This corresponds to the -Xasm_option=-Xprn_path option of the ccrh command. This property is displayed only when [Yes(-Xasm_option=-Xprn_path)] in the [Output assemble list file] property is selected.	
	Default	%BuildModeName%
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.
	Restriction	Up to 247 characters

(10) [MISRA-C:2004 Rule Check]

The detailed information on the MISRA-C:2004 rule check are displayed and the configuration can be changed.

Apply rule	Select the MISRA-C:2004 rules to be applied. This corresponds to the -Xmisra2004 option of the ccrh command.		
	Default	Not apply rule(None)	
	How to change	Select from the drop-down list.	
	Restriction	Apply all rules(-Xmisra2004=all)	Checks the source code against all of the rules which are supported.
		Apply specified rule number(-Xmisra2004=apply)	Checks the source code against the rules with the specified numbers among the rules which are supported.
		Ignore specified rule number(-Xmisra2004=ignore)	Checks the source code against the rules that do not match the specified numbers among the rules which are supported.
		Apply rules that are classified as "required"(-Xmisra2004=required)	Checks the source code against the rules of the "required" type.
		Apply rules that are classified as "required" and specified rule number(-Xmisra2004=required_add)	Checks the source code against the rules of the "required" type and the rules with the specified numbers among the rules which are supported.
		Ignore specified rule number from rules that are classified as "required"(-Xmisra2004=required_remove)	Checks the source code against the rules of the "required" type except for the rules with the specified numbers among the rules which are supported.
		Apply rules that are described in the specified file(-Xmisra2004=<file name>)	Checks the source code against the rules with the numbers described in specified file among the rules which are supported.
		Not apply rule(None)	Does not apply the MISRA-C:2004 rules.

Rule number description file	Specify the rule number description file (MISRA-C:2004 rule file). The following placeholders are supported. %BuildModeName%: Replaces with the build mode name. %MicomToolPath%: Replaces with the absolute path of the install folder of this product. %ProjectName%: Replaces with the project name. This corresponds to the -Xmisra2004 option of the ccrh command. This property is displayed only when [Apply rules that are described in the specified file(-Xmisra2004=<file name>)] in the [Apply rule] property is selected.	
	Default	Blank
	How to change	Directly enter in the text box or edit by the Specify MISRA-C:2004 Rule File dialog box which appears when clicking the [...] button.
	Restriction	Up to 259 characters
Rule number	Specify the rule number to be checked. Specify at least one rule number in decimal. This corresponds to the -Xmisra2004 option of the ccrh command. This property is displayed only when [Apply specified rule number(-Xmisra2004=apply)] in the [Apply rule] property is selected.	
	Default	Blank
	How to change	Directly enter in the text box or edit by the Specify Rule Number dialog box which appears when clicking the [...] button.
	Restriction	Up to 259 characters
Exclusion rule number	Specify the rule number to be excluded from the check. Specify at least one rule number in decimal. This corresponds to the -Xmisra2004 option of the ccrh command. This property is displayed only when [Ignore specified rule number(-Xmisra2004=ignore)] in the [Apply rule] property is selected.	
	Default	Blank
	How to change	Directly enter in the text box or edit by the Specify Rule Number dialog box which appears when clicking the [...] button.
	Restriction	Up to 259 characters
Check rule number besides required rule	Specify the rule number to be checked besides the required rules. Specify at least one rule number in decimal. This corresponds to the -Xmisra2004 option of the ccrh command. This property is displayed only when [Apply rules that are classified as "required" and specified rule number(-Xmisra2004=required_add)] in the [Apply rule] property is selected.	
	Default	Blank
	How to change	Directly enter in the text box or edit by the Specify Rule Number dialog box which appears when clicking the [...] button.
	Restriction	Up to 259 characters

<table>
<tr><td rowspan="4">Exclusion rule number from required rule</td><td colspan="2">Specify the required rule number to be excluded from the check.
Specify at least one rule number in decimal.
This corresponds to the -Xmisra2004 option of the ccrh command.
This property is displayed only when [Ignore specified rule number from rules that are classified as "required"(-Xmisra2004=required_remove)] in the [Apply rule] property is selected.</td></tr>
<tr><td>Default</td><td>Blank</td></tr>
<tr><td>How to change</td><td>Directly enter in the text box or edit by the Specify Rule Number dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td>Up to 259 characters</td></tr>
<tr><td rowspan="4">Rule check exclusion file</td><td colspan="2">Specify files that will not be checked against the MISRA-C:2004 rules.
The following placeholders are supported.
%BuildModeName%: Replaces with the build mode name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectName%: Replaces with the project name.
This corresponds to the -Xignore_files_misra option of the ccrh command.
This property is displayed only in the following cases.
- When [Apply all rules] is selected in the [Apply rule] property
- When [Apply rules that are classified as "required"] is selected in the [Apply rule] property
- When [Apply specified rule number] is selected in the [Apply rule] property and a rule number is specified in the [Rule number] property
- When [Ignore specified rule number] is selected in the [Apply rule] property and a rule number is specified in the [Rule number] property
- When [Apply rules that are classified as "required" and specified rule number] is selected in the [Apply rule] property and a rule number is specified in the [Check rule number besides required rule] property
- When [Ignore specified rule number from rules that are classified as "required"] is selected in the [Apply rule] property and a rule number is specified in the [Exclusion rule number from required rule] property
- When [Apply rules that are described in the specified file] is selected in the [Apply rule] property and a rule number description file is specified in the [Rule number description file] property</td></tr>
<tr><td>Default</td><td>Rule check exclusion file[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Path Edit dialog box which appears when clicking the [...] button.
-> Edit by the Add Excluding File dialog box which appears when clicking the [Browse...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 259 characters</td></tr>
</table>

<table>
<tr><td rowspan="5">Output message of the enhanced key word and extended specifications</td><td colspan="3">Select whether to output the message of the enhanced key word and extended specifications.
This corresponds to the -Xcheck_language_extention option of the ccrh command.
This property is displayed only in the following cases.
- When [Apply all rules] is selected in the [Apply rule] property
- When [Apply rules that are classified as "required"] is selected in the [Apply rule] property
- When [Apply specified rule number] is selected in the [Apply rule] property and a rule number is specified in the [Rule number] property
- When [Ignore specified rule number] is selected in the [Apply rule] property and a rule number is specified in the [Rule number] property
- When [Apply rules that are classified as "required" and specified rule number] is selected in the [Apply rule] property and a rule number is specified in the [Check rule number besides required rule] property
- When [Ignore specified rule number from rules that are classified as "required"] is selected in the [Apply rule] property and a rule number is specified in the [Exclusion rule number from required rule] property
- When [Apply rules that are described in the specified file] is selected in the [Apply rule] property and a rule number description file is specified in the [Rule number description file] property</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Restriction</td><td>Yes(-Xcheck_language_extension)</td><td>Enables MISRA-C:2004 rule check and outputs messages when the rule check is partially suppressed by the unique language specifications extended from the C language standard.</td></tr>
<tr><td>No</td><td>Disables MISRA-C:2004 rule check is disabled, which are partially suppressed by the extended language specifications.</td></tr>
</table>

(11) [Others]

Other detailed information on compilation is displayed and the configuration can be changed.

This category is displayed only when [No] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected.

<table>
<tr><td>Commands executed before compile processing</td><td colspan="2">Specify the command to be executed before compile processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%CompiledFile%: Replaces with the absolute path of the output file under compiling.
%InputFile%: Replaces with the absolute path of the file to be compiled.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%Options%: Replaces with the command line option under build execution.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%Program%: Replaces with the program name under execution.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed before compile processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.</td></tr>
<tr><td></td><td>Default</td><td>Commands executed before compile processing[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr>
</table>

<table>
<tr><td rowspan="5">Commands executed after compile processing</td><td colspan="2">Specify the command to be executed after compile processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%CompiledFile%: Replaces with the absolute path of the output file under compiling.
%InputFile%: Replaces with the absolute path of the file to be compiled.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%Options%: Replaces with the command line option under build execution.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%Program%: Replaces with the program name under execution.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed after compile processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Commands executed after compile processing[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr>
<tr><td rowspan="4">Other additional options</td><td colspan="2">Input the compile option to be added additionally.
The options set here are added at the end of the compile options group.</td></tr>
<tr><td>Default</td><td>Blank</td></tr>
<tr><td>How to change</td><td>Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td>Up to 259 characters</td></tr>
</table>

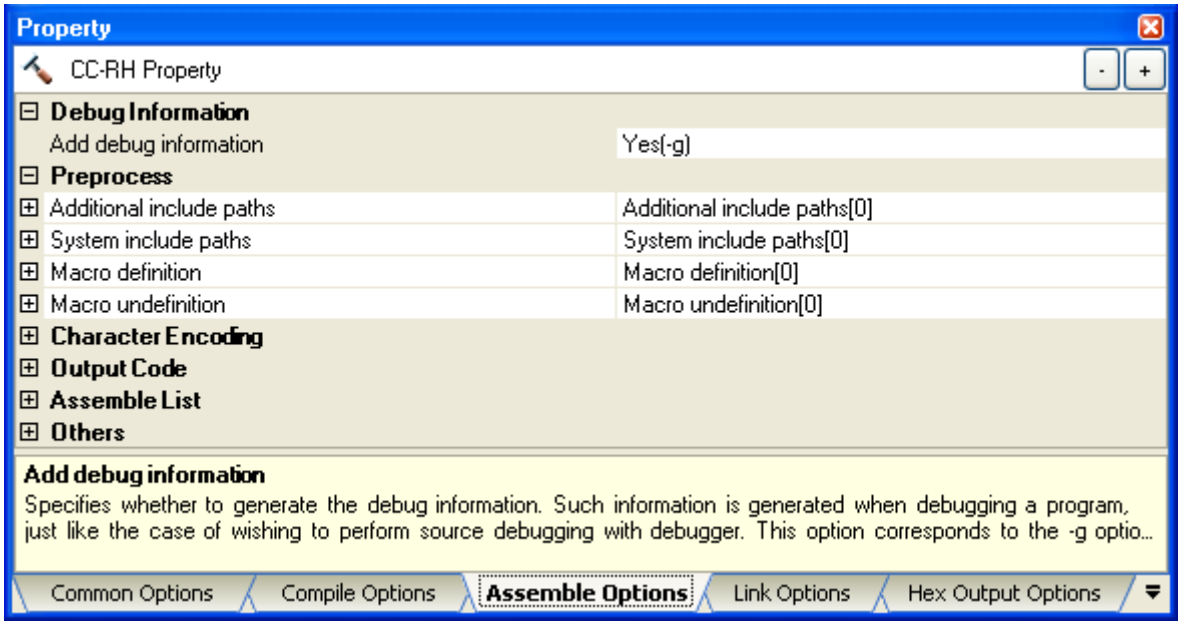
[Assemble Options] tab

This tab shows the detailed information on the assemble phase categorized by the following and the configuration can be changed.

(1) [Debug Information]
(2) [Preprocess]
(3) [Character Encoding]
(4) [Output Code]
(5) [Assemble List]
(6) [Others]

Caution This tab is displayed only when [No] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected.

Figure A-6. Property Panel: [Assemble Options] Tab



[Description of each category]

(1) [Debug Information]

The detailed information on debug information is displayed and the configuration can be changed.

Add debug information	Select whether to generate the debug information. It is possible to perform source debugging with the debugger by outputting information for source debugging to the output file. This corresponds to the -g option of the ccrh command.		
	Default	Yes(-g)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-g)	Generates the debug information.
		No	Does not generate the debug information.

(2) [Preprocess]

The detailed information on preprocessing is displayed and the configuration can be changed.

<table>
<tr><td>Additional include paths</td><td colspan="2">Specify the additional include paths during assembling.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
The specified include path is searched with higher priority than the standard include file folder of CC-RH.
The reference point of the path is the project folder.
When this property is omitted, only the standard folder of CC-RH is searched.
This corresponds to the -I option of the ccrh command.
The specified include path is displayed as the subproperty.
When the include path is added to the project tree, the path is added to the top of the subproperties.
Uppercase characters and lowercase characters are not distinguished for the include paths.</td></tr>
<tr><td></td><td>Default</td><td>Additional include paths[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td>Edit by the Path Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 259 characters
Up to 256 items can be specified.</td></tr>
</table>

<table>
<tr><td rowspan="4">System include paths</td><td colspan="2">Change the specified order of the include paths which the system set during assembling.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
The system include path is searched with lower priority than the additional include path.
The reference point of the path is the project folder.
This corresponds to the -I option of the ccrh command.
The include path is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>System include paths[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the System Include Path Order dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td>Changes not allowed (Only the specified order of the include paths can be changed.)</td></tr>
<tr><td rowspan="4">Macro definition</td><td colspan="2">Specify the name of the macro to be defined.
Specify in the format of "macro name=defined value", with one macro name per line.
The "=defined value" part can be omitted, and in this case, "1" is used as the defined value.
This corresponds to the -D option of the ccrh command.
The specified macro is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Macro definition[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 256 characters
Up to 256 items can be specified.</td></tr>
<tr><td rowspan="4">Macro undefinition</td><td colspan="2">Specify the macro name to be undefined.
Specify in the format of "macro name", with one macro name per line.
This corresponds to the -U option of the ccrh command.
The specified macro is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Macro undefinition[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 256 characters
Up to 256 items can be specified.</td></tr>
</table>

(3) [Character Encoding]

The detailed information on character encoding is displayed and the configuration can be changed.

Character encoding	Select the character code to be used for Japanese comments and character strings in the source file. This corresponds to the -Xcharacter_set option of the ccrh command.		
	Default	Auto(None)	
	How to change	Select from the drop-down list.	
	Restriction	Auto(None)	Interprets the Japanese character code in the source file as SJIS.
		SJIS(-Xcharacter_set=sjis)	Interprets the Japanese character code in the source file as SJIS.
		EUC(-Xcharacter_set=euc_jp)	Interprets the Japanese character code in the source file as EUC.
		UFT-8(-Xcharacter_set=utf8)	Interprets the Japanese character code in the source file as UFT-8.
		Big5(-Xcharacter_set=big5)	Interprets the Chinese character code in the source file as Traditional Chinese.
		GB2312(-Xcharacter_set=gb2312)	Interprets the Chinese character code in the source file as Simplified Chinese.
		No-process(-Xcharacter_set=none)	Does not interpret the Japanese/Chinese character code in the source file.

(4) [Output Code]

The detailed information on output codes is displayed and the configuration can be changed.

Use 32-bit branch instruction	Select whether to use the far jump function for the jarl and jr instructions. By using the far jump function, it is assumed that the jarl and jr instructions are jarl32 and jr32 instructions, and assembling is performed. This corresponds to the -Xasm_option=-Xasm_far_jump option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xasm_option=-Xasm_far_jump)	Assumes that the jarl and jr instructions are jarl32 and jr32 instructions, and performs assembling.
		No	Performs assembly as a jarl or jr instruction.

(5) [Assemble List]

The detailed information on the assemble list is displayed and the configuration can be changed.

Output assemble list file	Select whether to output the assemble list file. This corresponds to the -Xasm_option=-Xprn_path option of the ccrh command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xasm_option=-Xprn_path)	Outputs the assemble list file.
		No	Does not output the assemble list file.

Output folder for assemble list file	Specify the folder which the assemble list file is output. The assemble list file is output under the source file name with the extension replaced by ".prn". If a relative path is specified, the reference point of the path is the main project or subproject folder. If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different). The following placeholder is supported. %BuildModeName%: Replaces with the build mode name. If this is blank, it is assumed that the project folder has been specified. This corresponds to the -Xasm_option=-Xprn_path option of the ccrh command. This property is displayed only when [Yes(-Xasm_option=-Xprn_path)] in the [Output assemble list file] property is selected.	
	Default	%BuildModeName%
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.
	Restriction	Up to 247 characters

(6) [Others]

Other detailed information on assembly is displayed and the configuration can be changed.

<table>
<tr><td>Commands executed before assemble processing</td><td>Specify the command to be executed before assemble processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%AssembledFile%: Replaces with the absolute path of the output file under assembling.
%BuildModeName%: Replaces with the build mode name.
%InputFile%: Replaces with the absolute path of the file to be assembled.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%Options%: Replaces with the command line option under build execution.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%Program%: Replaces with the program name under execution.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed before assemble processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.
<table><tr><td>Default</td><td>Commands executed before assemble processing[number of defined items]</td></tr><tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr><tr><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr></table></td></tr>
</table>

<table>
<tr><td rowspan="4">Commands executed after assemble processing</td><td colspan="2">Specify the command to be executed after assemble processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%AssembledFile%: Replaces with the absolute path of the output file under assembling.
%BuildModeName%: Replaces with the build mode name.
%InputFile%: Replaces with the absolute path of the file to be assembled.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%Options%: Replaces with the command line option under build execution.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%Program%: Replaces with the program name under execution.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed after assemble processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Commands executed after assemble processing[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr>
<tr><td rowspan="4">Other additional options</td><td colspan="2">Input the assemble option to be added additionally.
The options set here are added at the end of the assemble options group.</td></tr>
<tr><td>Default</td><td>Blank</td></tr>
<tr><td>How to change</td><td>Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td>Up to 259 characters</td></tr>
</table>

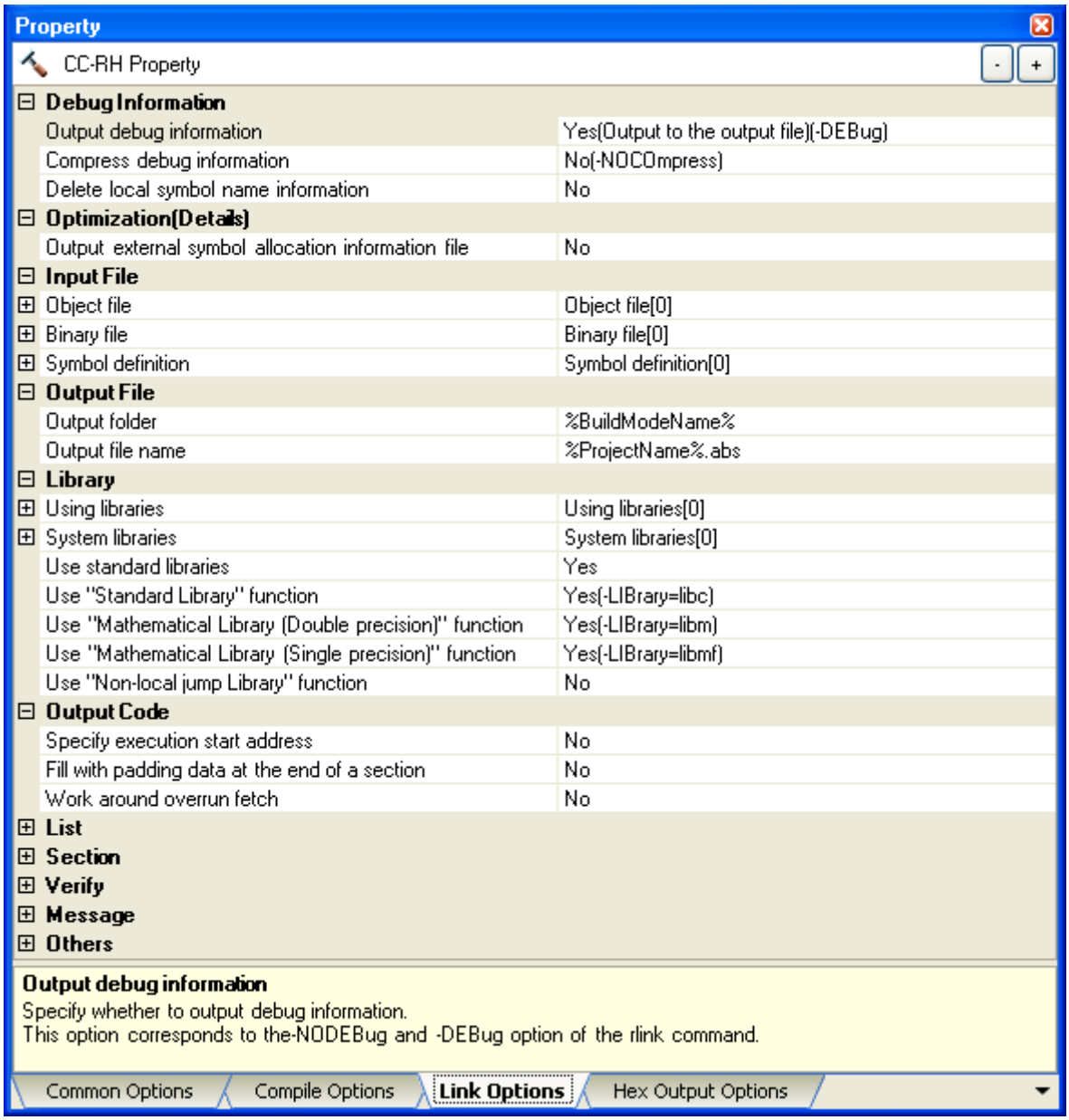
[Link Options] tab

This tab shows the detailed information on the link phase categorized by the following and the configuration can be changed.

(1) [Debug Information]
(2) [Optimization(Details)]
(3) [Input File]
(4) [Output File]
(5) [Library]
(6) [Output Code]
(7) [List]
(8) [Section]
(9) [Verify]
(10) [Message]
(11) [Others]

Caution This tab is not displayed for the library project.

Figure A-7. Property Panel: [Link Options] Tab



[Description of each category]

(1) [Debug Information]

The detailed information on debug information is displayed and the configuration can be changed.

Output debug information	Select whether to output debug information. This corresponds to the -DEBug and -NODEBug options of the rlink command.		
	Default	Yes(Output to the output file)(-DEBug)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(Output to the output file)(-DEBug)	Outputs debug information.
		No(-NODEBug)	Does not output debug information.
Compress debug information	Select whether to compress debug information. This corresponds to the -COmpress and -NOCOmpress options of the rlink command. This property is displayed only when [Yes(Output to the output file)(-DEBug)] in the [Output debug information] property is selected.		
	Default	No(-NOCOmpress)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-COmpress)	Compresses debug information. The loading speed of the debugger will be improved.
		No(-NOCOmpress)	Does not compress the debug information. The link time will be shorten.
Delete local symbol name information	Select whether to delete local symbol name information. This corresponds to the -Hide option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Hide)	Deletes information of the local symbol name.
		No	Does not delete information of the local symbol name.

(2) [Optimization(Details)]

The detailed information on the optimization is displayed and the configuration can be changed.

Output external symbol allocation information file	Select whether to output an external symbol allocation information file. This property is changed to [Yes(-Map)] when [Yes(Optimizes the inter-module)(-Omap)] in the [Optimize accesses to external variables] property in the [Optimization(Details)] category from the [Compile Options] tab is selected. This corresponds to the -MAp option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-MAp)	Outputs an external symbol allocation information file.
		No	Does not output an external symbol allocation information file.

(3) [Input File]

The detailed information on input files is displayed and the configuration can be changed.

<table>
<tr><td>Object file</td><td colspan="2">Specify the object files.
Specify in the format of "library(module)", with one entry name per line.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
This corresponds to the -Input option of the rlink command.
The object file name is displayed as the subproperty.</td></tr>
<tr><td></td><td>Default</td><td>Object file[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 1024 characters
Up to 256 items can be specified.</td></tr>
</table>

<table>
<tr><td rowspan="4">Binary file</td><td colspan="2">Specify the binary files.
Specify in the format of "file name(section name[:number of alignment][/section attribute][,symbol name])", with one entry per line.
[:number of alignment], [/section attribute], and [,symbol name] can be omitted.
The value that can be specified for number of alignment is 1, 2, 4, 8, 16, or 32.
If the specification is omitted, it is assumed that 1 has been specified.
"CODE" or "DATA" can be specified as section attribute.
If the specification is omitted, all attributes such as the ability to write, read, and execute, will be all valid.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
This corresponds to the -Binary option of the rlink command.
The binary file name is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Binary file[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 1024 characters
Up to 256 items can be specified.</td></tr>
<tr><td rowspan="4">Symbol definition</td><td colspan="2">Define the symbols.
Specify in the format of "symbol name=symbol name" or "symbol name=numerical value", with one entry name per line.
Specify the numerical value in hexadecimal.
This corresponds to the -DEFine option of the rlink command.
The symbol name is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Symbol definition[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 256 characters
Up to 256 items can be specified.</td></tr>
</table>

(4) [Output File]

The detailed information on output files is displayed and the configuration can be changed.

<table>
<tr><td>Output folder</td><td colspan="2">Specify the output folder.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
If this is blank, it is assumed that the project folder has been specified.
This corresponds to the -OUtput option of the rlink command.</td></tr>
<tr><td></td><td>Default</td><td>%BuildModeName%</td></tr>
<tr><td></td><td>How to change</td><td>Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 247 characters</td></tr>
<tr><td>Output file name</td><td colspan="2">Specify the output file name.
If the extension is omitted, ".abs" is automatically added.
The following placeholders are supported.
%ActiveProjectName%: Replaces with the active project name.
%MainProjectName%: Replaces with the main project name.
%ProjectName%: Replaces with the project name.
This corresponds to the -OUtput option of the rlink command.</td></tr>
<tr><td></td><td>Default</td><td>%ProjectName%.abs</td></tr>
<tr><td></td><td>How to change</td><td>Directly enter in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 259 characters</td></tr>
</table>

(5) [Library]

The detailed information on the library is displayed and the configuration can be changed.

<table>
<tr><td>Using libraries</td><td colspan="3">Specify the library files to be used.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
This corresponds to the -LIBrary option of the rlink command.
The library file name is displayed as the subproperty.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">Using libraries[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Edit by the Path Edit dialog box which appears when clicking the [...] button.
-> Edit by the Specify Using Library File dialog box which appears when clicking the [Browse...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td colspan="2">Up to 259 characters
Up to 65536 items can be specified.</td></tr>
<tr><td>System libraries</td><td colspan="3">The system library files are displayed.
This corresponds to the -LIBrary option of the rlink command.
The system library file name is displayed as the subproperty.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">System libraries[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Changes not allowed</td></tr>
<tr><td>Use standard libraries</td><td colspan="3">Select whether to use the standard libraries provided by the compiler.
This corresponds to the -LIBrary option of the rlink command.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td></td><td rowspan="2">Restriction</td><td>Yes</td><td>Uses the standard libraries.</td></tr>
<tr><td></td><td>No</td><td>Does not use the standard libraries.</td></tr>
<tr><td>Use "Standard Library" function</td><td colspan="3">Select whether to use the standard library functions.
This property is changed to [Yes(-LIBrary=libc)] when [Yes] in the [Use "Mathematical Library (Double precision)" function] or [Use "Mathematical Library (Single precision)" function] property is selected.
This corresponds to the -LIBrary option of the rlink command.
This property is displayed only when [Yes] in the [Use standard libraries] property is selected.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">Yes(-LIBrary=libc)</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td></td><td rowspan="2">Restriction</td><td>Yes(-LIBrary=libc)</td><td>Uses the standard library functions.</td></tr>
<tr><td></td><td>No</td><td>Does not use the standard library functions.</td></tr>
</table>

Use "Mathematical Library (Double precision)" function	Select whether to use the mathematical library (double precision) functions. This property is changed to [No] when [No] in the [Use standard libraries] property is selected. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	Yes(-LIBrary=libm)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libm)	Uses the mathematical library (double precision) functions.
		No	Does not use the mathematical library (double precision) functions.
Use "Mathematical Library (Single precision)" function	Select whether to use the mathematical library (single precision) functions. This property is changed to [No] when [No] in the [Use standard libraries] property is selected. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	Yes(-LIBrary=libmf)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libmf)	Uses the mathematical library (single precision) functions.
		No	Does not use the mathematical library (single precision) functions.
Use "Non-local jump Library" function	Select whether to use the non-local jump library functions. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libsetjmp)	Uses the non-local jump library functions.
		No	Does not use the non-local jump library functions.

(6) [Output Code]

The detailed information on output codes is displayed and the configuration can be changed.

Specify execution start address	Select whether to specify the execution start address with the external defined symbol or address. This corresponds to the -ENTry option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-ENTry)	Specifies the execution start address with the external defined symbol or address.
		No	Does not specify the execution start address with the external defined symbol or address.

Execution start address	Specify the execution start address. Specify in the format of "*symbol name*" or "*address*". Specify the address in hexadecimal. This corresponds to the -ENTry option of the rlink command. The execution start address is displayed as the subproperty. This property is displayed only when [Yes(-ENTry)] in the [Specify execution start address] property is selected.		
	Default	Blank	
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.	
	Restriction	Up to 32767 characters	
Fill with padding data at the end of a section	Select whether to fill with padding data at the end of a section. This corresponds to the -PADDING option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-PADDING)	Fills in data at the end of a section so that the section size is a multiple of the alignment of the section.
		No	Does not fill with padding data at the end of a section.
Work around overrun fetch	Select whether to prevent reading of vacant areas due to overrun fetch. This corresponds to the -OVERRUN_FETCH option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-OVERRUN_FETCH)	Inserts 128-byte NOP instructions in a vacant area when a 128-byte or larger vacant area exists between sections or immediately after the end code section.
		No	Does not prevent reading of vacant areas due to overrun fetch.

(7) [List]

The detailed information on the list is displayed and the configuration can be changed.

Output link map file	Select whether to output the link map file. This corresponds to the -LISt and -SHow options of the rlink command.		
	Default	Yes(List contents=specify)(-LISt)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(List contents=not specify)(-LISt -SHow)	Outputs information according to the output format to the link map file.
		Yes(List contents=ALL)(-LISt -SHow=ALL)	Outputs all information according to the output format to the link map file.
		Yes(List contents=specify)(-LISt)	Outputs the specified information to the link map file.
		No	Does not output the link map file.

Output symbol information	Select whether to output the symbol information (symbol address, size, type, and optimization contents). This corresponds to the -SHow=SYmbol option of the rlink command. This property is displayed only when [Yes(List contents=specify)(-LISt)] in the [Output link map file] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-SHow=SYmbol)	Outputs the symbol information.
		No	Does not output the symbol information.
Output number of symbol reference	Select whether to output the number of symbol references. This corresponds to the -SHow=Reference option of the rlink command. This property is displayed only when [Yes(List contents=specify)(-LISt)] in the [Output link map file] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-SHow=Reference)	Outputs the number of symbol references.
		No	Does not output the number of symbol references.
Output cross reference information	Select whether to output the cross reference information. This corresponds to the -SHow=XReference option of the rlink command. This property is displayed only when [Yes(List contents=specify)(-LISt)] in the [Output link map file] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-SHow=Xreference)	Outputs the cross reference information.
		No	Does not output the cross reference information.
Output total sizes of sections	Select whether to output the total size of sections. This corresponds to the -SHow=Total_size option of the rlink command. This property is displayed only when [Yes(List contents=specify)(-LISt)] in the [Output link map file] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-SHow=Total_size)	Outputs the total sizes of sections separately for ROM-allocated sections and RAM-allocated sections.
		No	Does not output the total size of sections.

(8) [Section]

The detailed information on the section is displayed and the configuration can be changed.

Section start address	Specify the start address of the section. This corresponds to the -STARt option of the rlink command.	
	Default	VECT,.const,.text,.data/00000000,RESET/01000000,.data.R,.bss,.stack.bss/FEBF8000
	How to change	Directly enter in the text box or edit by the Section Settings dialog box which appears when clicking the [...] button.
	Restriction	Up to 32767 characters
Section that includes startup function	Specify the section including the startup function. Specify one section name per line. When this property is specified, the startup symbol called from the multi-core boot loader is registered in the boot loader project. This corresponds to the -FSymbol option of the rlink command. The section name is displayed as the subproperty. This property is displayed only for the project set in the [Constituent application projects] property from the [Boot Loader] tab for the Boot loader node.	
	Default	Section that includes startup function[*number of defined items*]
	How to change	Edit by the Text Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.
	Restriction	Up to 32767 characters Up to 65535 items can be specified.
Section that outputs external defined symbols to the file	Specify the section whose external defined symbols are output to a file. Specify one section name per line. This corresponds to the -FSymbol option of the rlink command. The section name is displayed as the subproperty.	
	Default	Section that outputs external defined symbols to the file[*number of defined items*]
	How to change	Edit by the Text Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.
	Restriction	Up to 32767 characters Up to 65535 items can be specified.
Section alignment	Specify the name of the section to change the number of alignment to 16 bytes. Specify one section name per line. This corresponds to the -ALIGNED_SECTION option of the rlink command. The section name is displayed as the subproperty.	
	Default	Section alignment[*number of defined items*]
	How to change	Edit by the Text Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.
	Restriction	Up to 32767 characters Up to 65535 items can be specified.

ROM to RAM mapped section	Specify the section that maps symbols from ROM to RAM. Specify in the format of "*ROM section name=RAM section name*", with one section name per line. This corresponds to the -ROm option of the rlink command. The section name is displayed as the subproperty.	
	Default	ROM to RAM mapped section[*number of defined items*] ".data=.data.R" is specified in the subproperty.
	How to change	Edit by the Text Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.
	Restriction	Up to 32767 characters Up to 65535 items can be specified.

(9) [Verify]

The detailed information on verification is displayed and the configuration can be changed.

Check section larger than specified range of address	Select whether to check the consistency of the address to which the section is allocated. This corresponds to the -CPu option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-CPu)	Checks the consistency of the address to which the section is allocated.
	Default	No	Does not check the consistency of the address to which the section is allocated.
Address range of memory type	Specify the address range of the memory type. Specify in the format of "*memory type=start address-end address*", with one entry per line. Any of "ROm", "RAm", or "FIX" can be specified as *memory type*. Specify *start address* and *end address* in hexadecimal. This corresponds to the -CPu option of the rlink command. The address range of the memory type is displayed as the subproperty. This property is displayed only when [Yes(-CPu)] in the [Check section larger than specified range of address] property is selected.		
	Default	Address range of memory type[*number of defined items*]	
	How to change	Edit by the Text Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.	
	Restriction	Up to 32767 characters Up to 65535 items can be specified.	

(10) [Message]

The detailed information on messages is displayed and the configuration can be changed.

Enable information message output	Select whether to enable the output of information messages. This corresponds to the -Message and -NOMessage options of the rlink command.		
	Default	No(-NOMessage)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Message)	Outputs information messages.
		No(-NOMessage)	Suppresses the output of information messages.
Suppress number of information message	Specify the number of the information message of which output is to be suppressed. If multiple message numbers are specified, delimit them with "," (comma) (example: 4,200). Also, a range of message numbers can be specified using "-" (hyphen) (example:4,200-203,1300). This corresponds to the -NOMessage option of the rlink command. This property is displayed when [No(-NOMessage)] in the [Enable information message output] property is selected.		
	Default	Blank	
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.	
	Restriction	Up to 2048 characters	
Notify unused symbol	Select whether to notify the defined symbol that is not referenced. This corresponds to the -MSg_unused option of the rlink command. This property is displayed only when [Yes(-Message)] in the [Enable information message output] property is selected or the [Suppress number of information message] property is specified.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-MSg_unused)	Notifies the defined symbol that is not referenced.
		No	Does not notify the defined symbol that is not referenced.
Change warning message to information message	Select whether to change the type of warning messages to information. This corresponds to the -CHange_message option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(All)(-CHange_message=Information)	Changes the type of all warning messages to information.
		Yes(Specify message number)(-CHange_message=Information=<Message number>)	Specifies the number of warning message of which type is to be changed to information.
		No	Does not change the type of warning messages.

Number of warning message	Specify the number of the warning message. If multiple message numbers are specified, delimit them with "," (comma) (example: 4,200). Also, a range of message numbers can be specified using "-" (hyphen) (example:4,200-203,1300). This corresponds to the -CHange_message option of the rlink command. This property is displayed only when [Yes(Specify message number)(-CHange_message=Information=<Message number>)] in the [Change warning message to information message] property is selected.		
	Default	Blank	
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.	
	Restriction	Up to 2048 characters	
Change information message to warning message	Select whether to change the type of information messages to warning. This corresponds to the -CHange_message option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(All)(-CHange_message=Warning)	Changes the type of all information messages to warning.
		Yes(Specify message number)(-CHange_message=Warning=<Message number>)	Specifies the number of information message of which type is to be changed to warning.
		No	Does not change the type of information messages.
Number of information message	Specify the number of the information message. If multiple message numbers are specified, delimit them with "," (comma) (example: 4,200). Also, a range of message numbers can be specified using "-" (hyphen) (example:4,200-203,1300). This corresponds to the -CHange_message option of the rlink command. This property is displayed only when [Yes(Specify message number)(-CHange_message=Information=<Message number>)] in the [Change warning message to information message] property is selected.		
	Default	Blank	
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.	
	Restriction	Up to 2048 characters	
Change information and warning message to error message	Select whether to change the type of information and warning messages to error. This corresponds to the -CHange_message option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(All)(-CHange_message=Error)	Changes the type of all information and warning messages to error.
		Yes(Specify message number)(-CHange_message=Error=<Message number>)	Specifies the number of information or warning message of which type is to be changed to error.
		No	Does not change the type of information and warning messages.

Number of information and warning message	Specify the number of the information and warning message. If multiple message numbers are specified, delimit them with "," (comma) (example: 4,200). Also, a range of message numbers can be specified using "-" (hyphen) (example:4,200-203,1300). This corresponds to the -CHange_message option of the rlink command. This property is displayed only when [Yes(Specify message number)(-CHange_message=Error=<Message number>)] in the [Change warning message to information message] property is selected.		
	Default	Blank	
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.	
	Restriction	Up to 2048 characters	

(11) [Others]

Other detailed information on linking is displayed and the configuration can be changed.

Output stack information file	Select whether to output the stack information file. This corresponds to the -STACk option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-STACk)	Outputs the stack information file.
		No	Does not output the stack information file.
Reduce memory occupancy	Select whether to reduce the memory capacity. This corresponds to the -MEMory option of the rlink command. This property is displayed only in the following cases. - When [No(-NODEBug)] in the [Output debug information] property or [No(-NOCOmpress)] in the [Compress debug information] property in the [Debug Information] category is selected - When [No] in the [Output external symbol allocation information file] property in the [Optimization(Details)] category is selected - When any one of the conditions below is met. (1) When [No] in the [Output link map file] property in the [List] category is selected (2) When [Yes(List contents=not specify)(-LISt -SHow)] in the [Output link map file] property in the [List] category is selected (3) When [Yes(List contents=specify)(-LISt)] in the [Output link map file] property, [No] in the [Output number of symbol reference] property, and [No] in the [Output cross reference information] property in the [List] category are selected - When [No] in the [Output stack information file] property is selected		
	Default	No(-MEMory=High)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-MEMory=Low)	Reduces the memory capacity. Select this item if processing is slow because a large project is linked and the memory size occupied by the linker exceeds the available memory in the machine used.
		No(-MEMory=High)	Executes the same processing as usual.

<table>
<tr><td rowspan="4">Display total size of sections</td><td colspan="3">Select whether to display the total size of sections after the linking.
This corresponds to the -Total_size option of the rlink command.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td>Restriction</td><td>Yes(-Total_size)</td><td>Displays the total size of sections after the linking.</td></tr>
<tr><td></td><td>No</td><td>Does not display the total size of sections after the linking.</td></tr>
<tr><td rowspan="5">Display copyright information</td><td colspan="3">Select whether to display copyright information.
This corresponds to the -LOgo and -NOLOgo options of the rlink command.
This property is displayed only when [No] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected.</td></tr>
<tr><td>Default</td><td colspan="2">No(-NOLOgo)</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Restriction</td><td>Yes(-LOgo)</td><td>Displays copyright information.</td></tr>
<tr><td>No(-NOLOgo)</td><td>Suppresses the output of copyright information.</td></tr>
<tr><td rowspan="4">Commands executed before link processing</td><td colspan="3">Specify the command to be executed before link processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%LinkedFile%: Replaces with the absolute path of the output file under link processing.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%Options%: Replaces with the command line option under build execution.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%Program%: Replaces with the program name under execution.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed before link processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.
This property is displayed only when [No] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected.</td></tr>
<tr><td>Default</td><td colspan="2">Commands executed before link processing[number of defined items]</td></tr>
<tr><td>How to change</td><td colspan="2">Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td colspan="2">Up to 1023 characters
Up to 64 items can be specified.</td></tr>
</table>

<table>
<tr><td rowspan="5">Commands executed after link processing</td><td colspan="2">Specify the command to be executed after link processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%LinkedFile%: Replaces with the absolute path of the output file under link processing.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%Options%: Replaces with the command line option under build execution.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%Program%: Replaces with the program name under execution.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed after link processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.
This property is displayed only when [No] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected.</td></tr>
<tr><td>Default</td><td>Commands executed after link processing[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr>
<tr><td rowspan="4">Other additional options</td><td colspan="2">Input the link option to be added additionally.
The options set here are added at the end of the link options group.</td></tr>
<tr><td>Default</td><td>Blank</td></tr>
<tr><td>How to change</td><td>Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td>Up to 259 characters</td></tr>
</table>

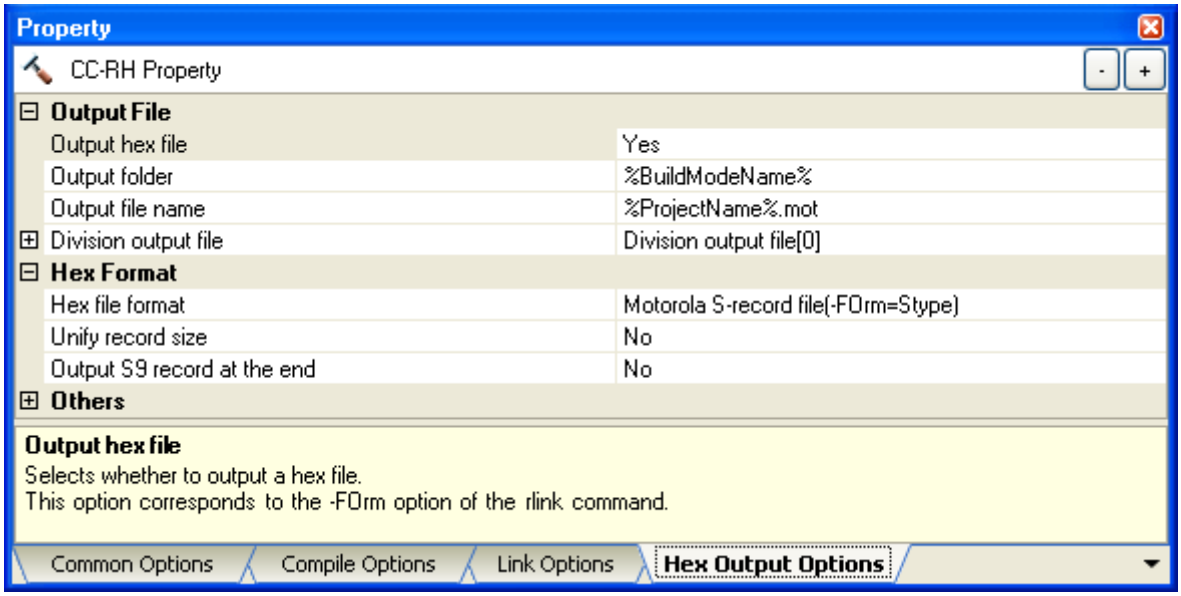
[Hex Output Options] tab

This tab shows the detailed information on the Hex output phase categorized by the following and the configuration can be changed.

(1) [Output File]
(2) [Hex Format]
(3) [Others]

Caution This tab is not displayed for the library project.

Figure A-8. Property Panel: [Hex Output Options] Tab



[Description of each category]

(1) [Output File]

The detailed information on output files is displayed and the configuration can be changed.

Output hex file	Select whether to output the hex file. This corresponds to the -FOrm option of the rlink command.		
	Default	Yes	
	How to change	Select from the drop-down list.	
	Restriction	Yes	Outputs the hex file.
	Default	No	Does not output the hex file.

<table>
<tr><td>Output folder</td><td colspan="2">Specify the folder which the hex file is output.
If a relative path is specified, the reference point of the path is the main project or subproject folder.
If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
If this is blank, it is assumed that the project folder has been specified.
This corresponds to the -OUtput option of the rlink command.
This property is displayed only when [Yes] in the [Output hex file] property is selected.</td></tr>
<tr><td></td><td>Default</td><td>%BuildModeName%</td></tr>
<tr><td></td><td>How to change</td><td>Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 247 characters</td></tr>
<tr><td>Output file name</td><td colspan="2">Specify the hex file name.
Be sure to specify this property.
If the extension is omitted, it is automatically added according to the selection in the [Hex file format] property in the [Hex Format] category.
When [Intel HEX file(-FOrm=Hexadecimal)] is selected: .hex
When [Motorola S-record file(-FOrm=Stype)] is selected: .mot
When [Binary file(-FOrm=Binary)] is selected: .bin
The following placeholders are supported.
%ActiveProjectName%: Replaces with the active project name.
%MainProjectName%: Replaces with the main project name.
%ProjectName%: Replaces with the project name.
This corresponds to the -OUtput option of the rlink command.
This property is displayed only when [Yes] in the [Output hex file] property is selected.</td></tr>
<tr><td></td><td>Default</td><td>%ProjectName%.mot</td></tr>
<tr><td></td><td>How to change</td><td>Directly enter in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 259 characters</td></tr>
</table>

<table>
<tr><td>Division output file</td><td colspan="3">Specify the division output files.
Specify in the format of "file name=start address-end address" (start address, end address: The start address and end address of the output range) or "file name=section name" (section name: The name of the output section), with one entry per line.
If multiple section names are specified, delimit them with a colon as in "file name=section name:section name" (example: file1.mot=sec1:sec2).
Specify the address in hexadecimal (example: file2.mot=400-4ff).
If the extension is omitted, it is automatically added according to the selection in the [Hex file format] property in the [Hex Format] category.
When [Intel HEX file(-FOrm=Hexadecimal)] is selected: .hex
When [Motorola S-record file(-FOrm=Stype)] is selected: .mot
When [Binary file(-FOrm=Binary)] is selected: .bin
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
This corresponds to the -OUtput option of the rlink command.
The division output file name is displayed as the subproperty.
This property is displayed only when [Yes] in the [Output hex file] property is selected.</td></tr>
<tr><td>Default</td><td colspan="2">Division output file[number of defined items]</td></tr>
<tr><td>How to change</td><td colspan="2">Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td colspan="2">Up to 259 characters
Up to 65535 items can be specified.</td></tr>
<tr><td>Use object uniting function</td><td colspan="3">Select whether to combine the hex files of the projects specified as constituent applications into a file.
This corresponds to the -FOrm option of the rlink command.
This property is displayed only in the following cases.
- For the multi-core boot loader project
- When [Yes] is selected in the [Output hex file] property and the same item is selected in the [Hex file format] property in the [Hex Format] category for the boot loader project and all component application projects.</td></tr>
<tr><td>Default</td><td colspan="2">No</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Restriction</td><td>Yes</td><td>Combines the hex files of the projects specified as constituent applications into a file.</td></tr>
<tr><td>No</td><td>Outputs the hex file for each project.</td></tr>
</table>

Output folder for united hex file	Specify the folder which the combined hex file is output. The following placeholders are supported. %ActiveProjectDir%: Replaces with the absolute path of the active project folder. %ActiveProjectName%: Replaces with the active project name. %BuildModeName%: Replaces with the build mode name. %MainProjectDir%: Replaces with the absolute path of the main project folder. %MainProjectName%: Replaces with the main project name. %MicomToolPath%: Replaces with the absolute path of the install folder of this product. %ProjectDir%: Replaces with the absolute path of the project folder. %ProjectName%: Replaces with the project name. %TempDir%: Replaces with the absolute path of the temporary folder. %WinDir%: Replaces with the absolute path of the Windows system folder. If this is blank, it is assumed that the project folder has been specified. This corresponds to the -OUtput option of the rlink command. This property is displayed only when [Yes] in the [Use object uniting function] property is selected.	
	Default	%BuildModeName%_merged
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.
	Restriction	Up to 247 characters

(2) [Hex Format]

The detailed information on the hex format is displayed and the configuration can be changed.
This category is displayed only when [Yes] in the [Output hex file] property in the [Output File] category is selected.

Hex file format	Select the format of the hex file to be output. This corresponds to the -FOrm option of the rlink command.		
	Default	Motorola S-record file(-FOrm=Stype)	
	How to change	Select from the drop-down list.	
	Restriction	Intel HEX file(-FOrm=Hexadecimal)	Outputs an Intel HEX file.
		Motorola S-record file(-FOrm=Stype)	Outputs a Motorola S-record file.
		Binary file(-FOrm=Binary)	Outputs a binary file.
Unify record size [Intel HEX file]	Select whether to output a specified data record regardless of the address range. This corresponds to the -RECord option of the rlink command. This property is displayed only when [Intel HEX file(-FOrm=Hexadecimal)] in the [Hex file format] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(Intel hex record)(-REcord=H16)	Outputs the Intel hex record.
		Yes(Intel expanded hex record)(-REcord=H20)	Outputs the Intel expanded hex record.
		Yes(Intel 32-bit hex record)(-REcord=H32)	Outputs the Intel 32-bit hex record.
		No	Outputs various data records according to each address.

Unify record size [Motorola S-record file]	Select whether to output a specified data record regardless of the address range. This corresponds to the -RECord option of the rlink command. This property is displayed only when [Motorola S-record file(-FOrm=Stype)] in the [Hex file format] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(S1 record)(-REcord=S1)	Outputs the S1 record.
		Yes(S2 record)(-REcord=S2)	Outputs the S2 record.
		Yes(S3 record)(-REcord=S3)	Outputs the S3 record.
		No	Outputs various data records according to each address.
Fill unused areas in the output ranges with the value	Select whether to fill the vacant area of the output range with data. This corresponds to the -SPace option of the rlink command. This property is displayed only when a file is specified in the [Division output file] property in the [Output File] category.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(Random)(-SPace=Random)	Fills the vacant area with random numbers.
		Yes(Specification value)(-SPace=<Numerical value>)	Fills the vacant area with the specified hexadecimal value.
		No	Does not fills the vacant area.
Output padding data	Specify the hexadecimal value to fill the vacant area. This corresponds to the -SPace option of the rlink command. This property is displayed only when [Yes(Specification value)(-SPace=<Numerical value>)] in the [Fill unused areas in the output ranges with the value] property is selected.		
	Default	FF	
	How to change	Directly enter in the text box.	
	Restriction	0 to FFFFFFFF (hexadecimal number)	
Specify byte count for data record	Select whether to specify the maximum byte count for a data record. This corresponds to the -BYte_count option of the rlink command. This property is displayed only when [Intel HEX file(-FOrm=Hexadecimal)] in the [Hex file format] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-BYte_count)	Specifies the maximum byte count for a data record.
		No	Specifies 0xFF as the maximum byte count for a data record.

Maximum byte count for data record	Specify the maximum byte count for a data record. This corresponds to the -BYte_count option of the rlink command. This property is displayed only when [Yes(-BYte_count)] in the [Specify byte count for data record] property is selected.		
	Default	FF	
	How to change	Directly enter to the text box.	
	Restriction	1 to FF (hexadecimal number)	
Output S9 record at the end	Select whether to output the S9 record at the end. This corresponds to the -S9 option of the rlink command. This property is displayed only when [Motorola S-record file(-FOrm=Stype)] in the [Hex file format] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-S9)	Outputs the S9 record at the end.
		No	Does not output the S9 record at the end.

(3) [Others]

Other detailed information on the hex output is displayed and the configuration can be changed.

This category is displayed only when [Yes] in the [Output hex file] property in the [Output File] category is selected.

Other additional options	Input the hex output options to be added additionally. The options set here are added at the end of the hex output options group.	
	Default	Blank
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.
	Restriction	Up to 259 characters

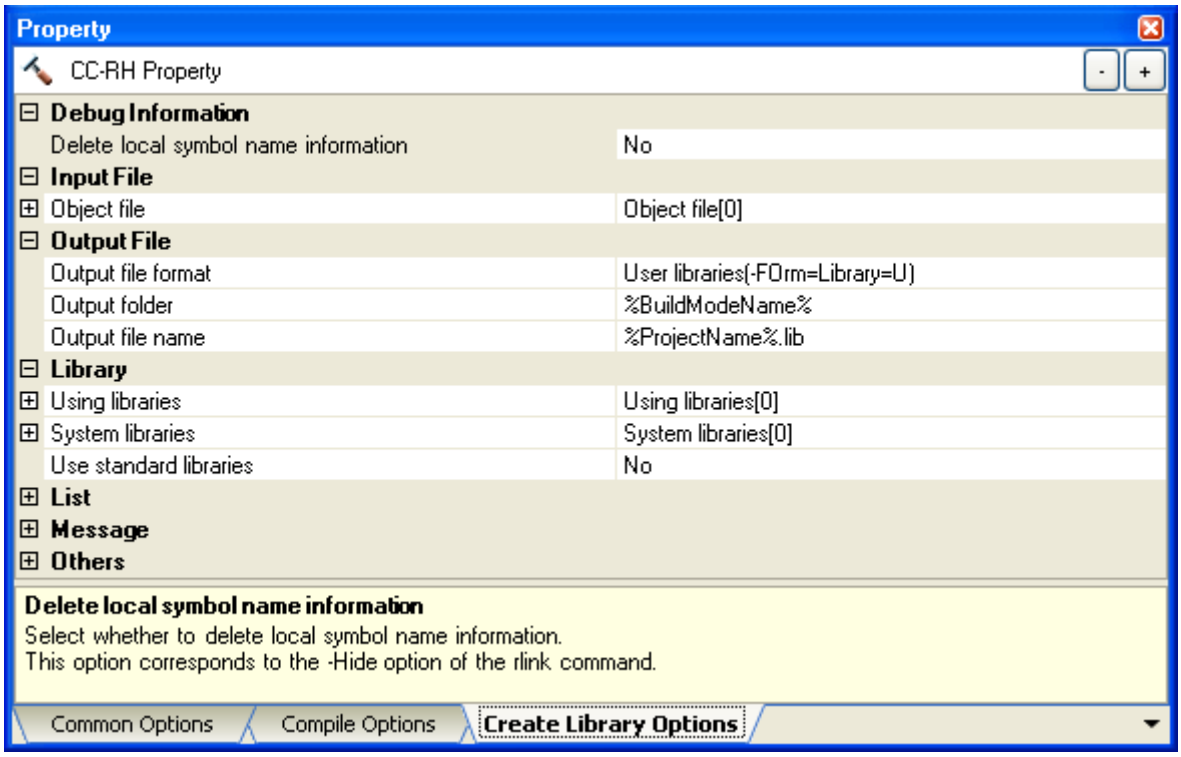
[Create Library Options] tab

This tab shows the detailed information on the create library phase categorized by the following and the configuration can be changed.

(1) [Debug Information]
(2) [Input File]
(3) [Output File]
(4) [Library]
(5) [List]
(6) [Message]
(7) [Others]

Caution This tab is displayed for the library project.

Figure A-9. Property Panel: [Create Library Options] Tab



[Description of each category]

(1) [Debug Information]

The detailed information on debug information is displayed and the configuration can be changed.

Output debug information	Select whether to output debug information. This corresponds to the -DEBug and -NODEBug options of the rlink command. This property is displayed only when [Relocatable file(-FOrm=Relocate)] in the [Output file format] property in the [Output File] category.		
	Default	Yes(Output to the output file)(-DEBug)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(Output to the output file)(-DEBug)	Outputs debug information.
		No	Does not output debug information.
Delete local symbol name information	Select whether to delete local symbol name information. This corresponds to the -Hide option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Hide)	Deletes information of the local symbol name.
		No	Does not delete information of the local symbol name.

(2) [Input File]

The detailed information on input files is displayed and the configuration can be changed.

Object file	Specify the object files. Specify in the format of "*library*(*module*)", with one entry name per line. The following placeholders are supported. %ActiveProjectDir%: Replaces with the absolute path of the active project folder. %ActiveProjectName%: Replaces with the active project name. %BuildModeName%: Replaces with the build mode name. %MainProjectDir%: Replaces with the absolute path of the main project folder. %MainProjectName%: Replaces with the main project name. %MicomToolPath%: Replaces with the absolute path of the install folder of this product. %ProjectDir%: Replaces with the absolute path of the project folder. %ProjectName%: Replaces with the project name. %TempDir%: Replaces with the absolute path of the temporary folder. %WinDir%: Replaces with the absolute path of the Windows system folder. This corresponds to the -Input option of the rlink command. The object file name is displayed as the subproperty.	
	Default	Object file[*number of defined items*]
	How to change	Edit by the Text Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.
	Restriction	Up to 1024 characters Up to 256 items can be specified.

Binary file	Specify the binary files. Specify in the format of "*file name*(*section name*[:*number of alignment*][/*section attribute*][,*symbol name*])", with one entry per line. [:*number of alignment*], [/*section attribute*], and [,*symbol name*] can be omitted. The value that can be specified for *number of alignment* is 1, 2, 4, 8, 16, or 32. If the specification is omitted, it is assumed that 1 has been specified. "CODE" or "DATA" can be specified as *section attribute*. If the specification is omitted, all attributes such as the ability to write, read, and execute, will be all valid. The following placeholders are supported. %ActiveProjectDir%: Replaces with the absolute path of the active project folder. %ActiveProjectName%: Replaces with the active project name. %BuildModeName%: Replaces with the build mode name. %MainProjectDir%: Replaces with the absolute path of the main project folder. %MainProjectName%: Replaces with the main project name. %MicomToolPath%: Replaces with the absolute path of the install folder of this product. %ProjectDir%: Replaces with the absolute path of the project folder. %ProjectName%: Replaces with the project name. %TempDir%: Replaces with the absolute path of the temporary folder. %WinDir%: Replaces with the absolute path of the Windows system folder. This corresponds to the -Binary option of the rlink command. The binary file name is displayed as the subproperty. This property is displayed only when [Relocatable file(-FOrm=Relocate)] in the [Output file format] property in the [Output File] category.	
	Default	Binary file[*number of defined items*]
	How to change	Edit by the Text Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.
	Restriction	Up to 1024 characters Up to 256 items can be specified.

(3) [Output File]

The detailed information on output files is displayed and the configuration can be changed.

Output file format	Select the format of the output file. This corresponds to the -FOrm option of the rlink command.		
	Default	User libraries(-FOrm=Library=U)	
	How to change	Select from the drop-down list.	
	Restriction	User libraries(-FOrm=Library=U)	Outputs a user library file.
		System libraries(-FOrm=Library=S)	Outputs a system library file.
		Relocatable file(-FOrm=Relocate)	Outputs a relocatable file.

<table>
<tr><td>Output folder</td><td colspan="2">Specify the output folder.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
If this is blank, it is assumed that the project folder has been specified.
This corresponds to the -OUtput option of the rlink command.</td></tr>
<tr><td></td><td>Default</td><td>%BuildModeName%</td></tr>
<tr><td></td><td>How to change</td><td>Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 247 characters</td></tr>
<tr><td>Output file name</td><td colspan="2">Specify the output file name.
If the extension is omitted, it is automatically added according to the selection in the [Output file format] property.
When [User libraries(-FOrm=Library=U)] is selected: .lib
When [System libraries(-FOrm=Library=S)] is selected: .lib
When [Relocatable file(-FOrm=Relocate)] is selected: .rel
The following placeholders are supported.
%ActiveProjectName%: Replaces with the active project name.
%MainProjectName%: Replaces with the main project name.
%ProjectName%: Replaces with the project name.
This corresponds to the -OUtput option of the rlink command.</td></tr>
<tr><td></td><td>Default</td><td>%ProjectName%.lib</td></tr>
<tr><td></td><td>How to change</td><td>Directly enter in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td>Up to 259 characters</td></tr>
</table>

(4) [Library]

The detailed information on the library is displayed and the configuration can be changed.

<table>
<tr><td>Using libraries</td><td colspan="3">Specify the library files to be used.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
This corresponds to the -LIBrary option of the rlink command.
The library file name is displayed as the subproperty.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">Using libraries[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Edit by the Path Edit dialog box which appears when clicking the [...] button.
-> Edit by the Specify Using Library File dialog box which appears when clicking the [Browse...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td colspan="2">Up to 259 characters
Up to 65536 items can be specified.</td></tr>
<tr><td>System libraries</td><td colspan="3">The system library files are displayed.
This corresponds to the -LIBrary option of the rlink command.
The system library file name is displayed as the subproperty.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">System libraries[number of defined items]</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Changes not allowed</td></tr>
<tr><td>Use standard libraries</td><td colspan="3">Select whether to use the standard libraries provided by the compiler.
This corresponds to the -LIBrary option of the rlink command.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">No</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td></td><td rowspan="2">Restriction</td><td>Yes</td><td>Uses the standard libraries.</td></tr>
<tr><td></td><td>No</td><td>Does not use the standard libraries.</td></tr>
<tr><td>Use "Standard Library" function</td><td colspan="3">Select whether to use the standard library functions.
This property is changed to [Yes(-LIBrary=libc)] when [Yes] in the [Use "Mathematical Library (Double precision)" function] or [Use "Mathematical Library (Single precision)" function] property is selected.
This corresponds to the -LIBrary option of the rlink command.
This property is displayed only when [Yes] in the [Use standard libraries] property is selected.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">No</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td></td><td rowspan="2">Restriction</td><td>Yes(-LIBrary=libc)</td><td>Uses the standard library functions.</td></tr>
<tr><td></td><td>No</td><td>Does not use the standard library functions.</td></tr>
</table>

Use "Mathematical Library (Double precision)" function	Select whether to use the mathematical library (double precision) functions. This property is changed to [No] when [No] in the [Use standard libraries] property is selected. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libm)	Uses the mathematical library (double precision) functions.
		No	Does not use the mathematical library (double precision) functions.
Use "Mathematical Library (Single precision)" function	Select whether to use the mathematical library (single precision) functions. This property is changed to [No] when [No] in the [Use standard libraries] property is selected. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libmf)	Uses the mathematical library (single precision) functions.
		No	Does not use the mathematical library (single precision) functions.
Use "Non-local jump Library" function	Select whether to use the non-local jump library functions. This corresponds to the -LIBrary option of the rlink command. This property is displayed only when [Yes] in the [Use standard libraries] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LIBrary=libsetjmp)	Uses the non-local jump library functions.
		No	Does not use the non-local jump library functions.

(5) [List]

The detailed information on the list is displayed and the configuration can be changed.

Output link map file	Select whether to output the library list file. This corresponds to the -LISt and -SHow options of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(List contents=not specify)(-LISt -SHow)	Outputs information according to the output format to the library list file.
		Yes(List contents=ALL)(-LISt -SHow=ALL)	Outputs all information according to the output format to the library list file.
		Yes(List contents=specify)(-LISt)	Outputs the specified information to the library list file.
		No	Does not output the library list file.

Output symbol information	Select whether to output the symbol information (symbol names within a module). This corresponds to the -SHow=SYmbol option of the rlink command. This property is displayed only when [Yes(List contents=specify)(-LISt)] in the [Output link map file] property is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-SHow=SYmbol)	Outputs the symbol information.
		No	Does not output the symbol information.
Output section list in a module	Select whether to output the list of the section names within the module. This corresponds to the -SHow=SEction option of the rlink command. This property is displayed only when [Yes(List contents=specify)(-LISt)] in the [Output link map file] property is selected and [User libraries(-FOrm=Library=U)] or [System libraries(-FOrm=Library=S)] in the [Output file format] propertry in the [Output File] category is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-SHow=SEction)	Outputs the list of the section names within the module.
		No	Does not output the list of the section names within the module.
Output cross reference information	Select whether to output the cross reference information. This corresponds to the -SHow=XReference option of the rlink command. This property is displayed only when [Yes(List contents=specify)(-LISt)] in the [Output link map file] property and [Relocatable file(-FOrm=Relocate)] in the [Output file format] propertry in the [Output File] category are selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-SHow=Xreference)	Outputs the cross reference information.
		No	Does not output the cross reference information.
Output total sizes of sections	Select whether to output the total size of sections. This corresponds to the -SHow=Total_size option of the rlink command. This property is displayed only when [Yes(List contents=specify)(-LISt)] in the [Output link map file] property and [Relocatable file(-FOrm=Relocate)] in the [Output file format] propertry in the [Output File] category are selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-SHow=Total_size)	Outputs the total sizes of sections separately for ROM-allocated sections and RAM-allocated sections.
		No	Does not output the total size of sections.

(6) [Message]

The detailed information on messages is displayed and the configuration can be changed.

Enable information message output	Select whether to enable the output of information messages. This corresponds to the -Message and -NOMessage options of the rlink command.		
	Default	No(-NOMessage)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Message)	Outputs information messages.
		No(-NOMessage)	Suppresses the output of information messages.
Suppress number of information message	Specify the number of the information message of which output is to be suppressed. If multiple message numbers are specified, delimit them with "," (comma) (example: 4,200). Also, a range of message numbers can be specified using "-" (hyphen) (example:4,200-203,1300). This corresponds to the -NOMessage option of the rlink command. This property is displayed when [No(-NOMessage)] in the [Enable information message output] property is selected.		
	Default	Blank	
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.	
	Restriction	Up to 2048 characters	
Change warning message to information message	Select whether to change the type of warning messages to information. This corresponds to the -CHange_message option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(All)(-CHange_message=Information)	Changes the type of all warning messages to information.
		Yes(Specify message number)(-CHange_message=Information=<Message number>)	Specifies the number of warning message of which type is to be changed to information.
		No	Does not change the type of warning messages.
Number of warning message	Specify the number of the warning message. If multiple message numbers are specified, delimit them with "," (comma) (example: 4,200). Also, a range of message numbers can be specified using "-" (hyphen) (example:4,200-203,1300). This corresponds to the -CHange_message option of the rlink command. This property is displayed only when [Yes(Specify message number)(-CHange_message=Information=<Message number>)] in the [Change warning message to information message] property is selected.		
	Default	Blank	
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.	
	Restriction	Up to 2048 characters	

Change information message to warning message	Select whether to change the type of information messages to warning. This corresponds to the -CHange_message option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(All)(-CHange_message=Warning)	Changes the type of all information messages to warning.
		Yes(Specify message number)(-CHange_message=Warning=<Message number>)	Specifies the number of information message of which type is to be changed to warning.
		No	Does not change the type of information messages.
Number of information message	Specify the number of the information message. If multiple message numbers are specified, delimit them with "," (comma) (example: 4,200). Also, a range of message numbers can be specified using "-" (hyphen) (example:4,200-203,1300). This corresponds to the -CHange_message option of the rlink command. This property is displayed only when [Yes(Specify message number)(-CHange_message=Information=<Message number>)] in the [Change warning message to information message] property is selected.		
	Default	Blank	
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.	
	Restriction	Up to 2048 characters	
Change information and warning message to error message	Select whether to change the type of information and warning messages to error. This corresponds to the -CHange_message option of the rlink command.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(All)(-CHange_message=Error)	Changes the type of all information and warning messages to error.
		Yes(Specify message number)(-CHange_message=Error=<Message number>)	Specifies the number of information or warning message of which type is to be changed to error.
		No	Does not change the type of information and warning messages.
Number of information and warning message	Specify the number of the information and warning message. If multiple message numbers are specified, delimit them with "," (comma) (example: 4,200). Also, a range of message numbers can be specified using "-" (hyphen) (example:4,200-203,1300). This corresponds to the -CHange_message option of the rlink command. This property is displayed only when [Yes(Specify message number)(-CHange_message=Error=<Message number>)] in the [Change warning message to information message] property is selected.		
	Default	Blank	
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.	
	Restriction	Up to 2048 characters	

(7) [Others]

Other detailed information on creating a library is displayed and the configuration can be changed.

Reduce memory occupancy	Select whether to reduce the memory capacity. This corresponds to the -MEMory option of the rlink command. This property is displayed only in the following cases. - When [No] in the [Delete local symbol name information] property in the [Debug Information] category is selected - When [User libraries(-FOrm=Library=U)] or [System libraries(-FOrm=Library=S)] in the [Output file format] property in the [Output File] category is selected		
	Default	No(-MEMory=High)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-MEMory=Low)	Reduces the memory capacity. Select this item if processing is slow because a large project is linked and the memory size occupied by the linker exceeds the available memory in the machine used.
		No(-MEMory=High)	Executes the same processing as usual.
Display total size of sections	Select whether to display the total size of sections after the linking. This corresponds to the -Total_size option of the rlink command. This property is displayed only when [Relocatable file(-FOrm=Relocate)] in the [Output file format] property in the [Output File] category.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Total_size)	Displays the total size of sections after the linking.
		No	Does not display the total size of sections after the linki4ng.
Display copyright information	Select whether to display copyright information. This corresponds to the -LOgo and -NOLOgo options of the rlink command. This property is displayed only when [No] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected.		
	Default	No(-NOLOgo)	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-LOgo)	Displays copyright information.
		No(-NOLOgo)	Suppresses the output of copyright information.

<table>
<tr><td>Commands executed before create library processing</td><td>Specify the command to be executed before library generation processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%LibraryFile%: Replaces with the absolute path of the output file under the library generation processing.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%Options%: Replaces with the command line option under build execution.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%Program%: Replaces with the program name under execution.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed before library generation processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.
This property is displayed only when [No] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected.
<table><tr><td>Default</td><td>Commands executed before library generate processing[number of defined items]</td></tr><tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr><tr><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr></table></td></tr>
</table>

<table>
<tr><td rowspan="5">Commands executed after create library processing</td><td colspan="2">Specify the command to be executed after library generation processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%LibraryFile%: Replaces with the absolute path of the output file under the library generation processing.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%Options%: Replaces with the command line option under build execution.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%Program%: Replaces with the program name under execution.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed after library generation processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.
This property is displayed only when [No] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected.</td></tr>
<tr><td>Default</td><td>Commands executed after library generate processing[number of defined items]</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr>
<tr><td rowspan="4">Other additional options</td><td colspan="2">Input the create library options to be added additionally.
The options set here are added at the end of the create library options group.</td></tr>
<tr><td>Default</td><td>Blank</td></tr>
<tr><td>How to change</td><td>Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td>Up to 259 characters</td></tr>
</table>

[Build Settings] tab

This tab shows the detailed information on each C source file, assembly source file, object file, and library file categorized by the following and the configuration can be changed.

(1) [Build]

Figure A-10. Property Panel: [Build Settings] Tab (When Selecting C Source File)

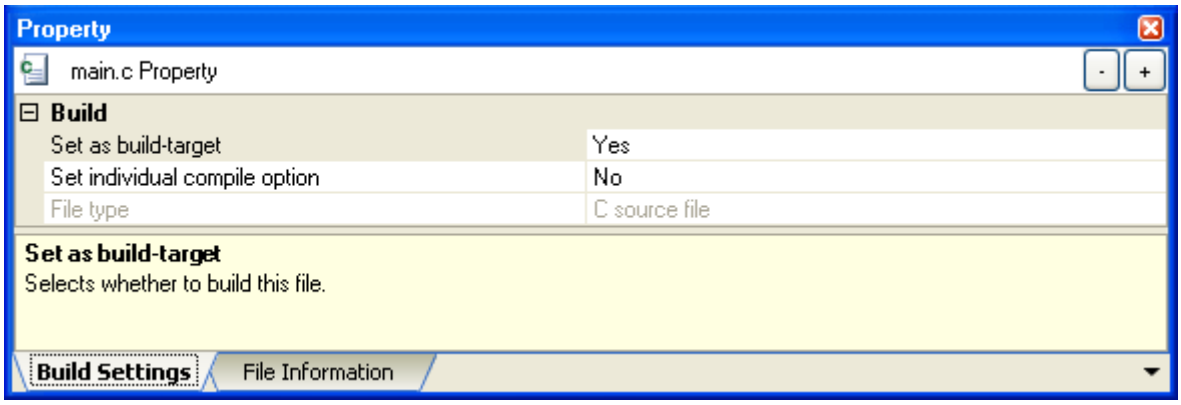


Figure A-11. Property Panel: [Build Settings] Tab (When Selecting Assembly Source File)

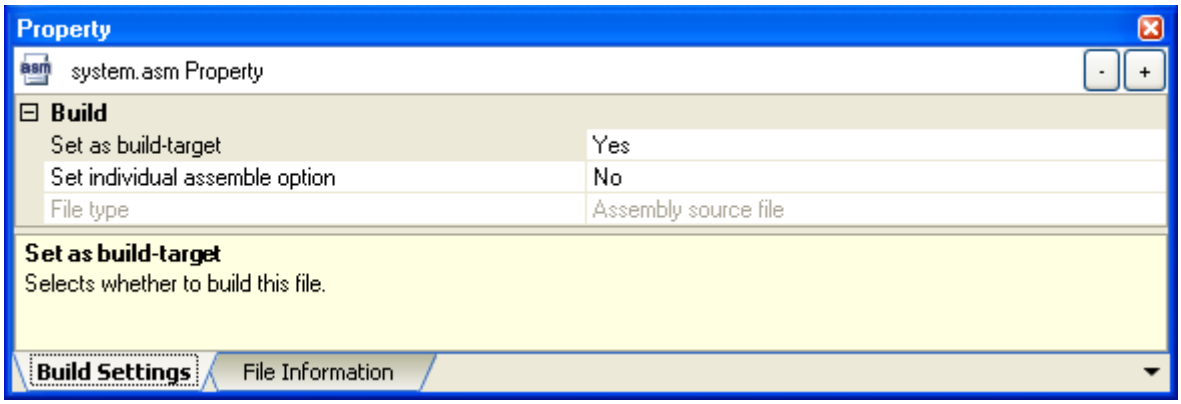


Figure A-12. Property Panel: [Build Settings] Tab (When Selecting Object File)

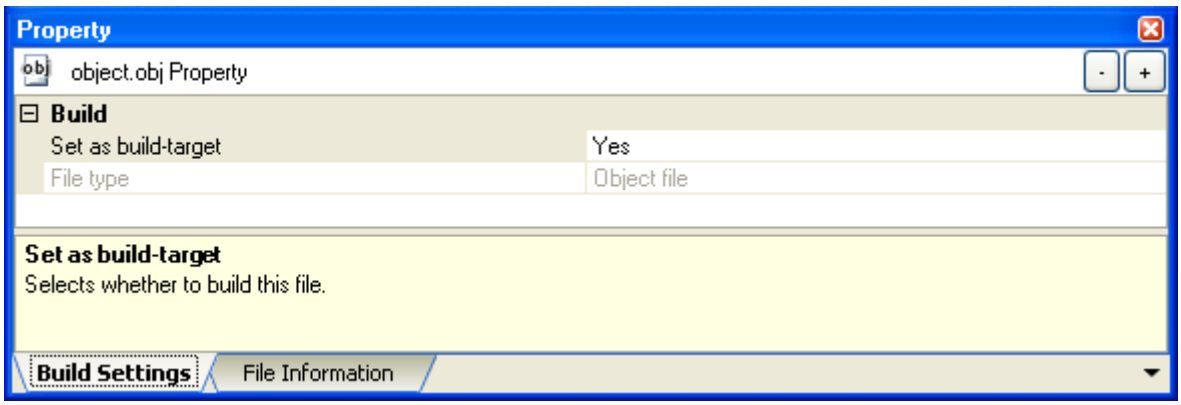
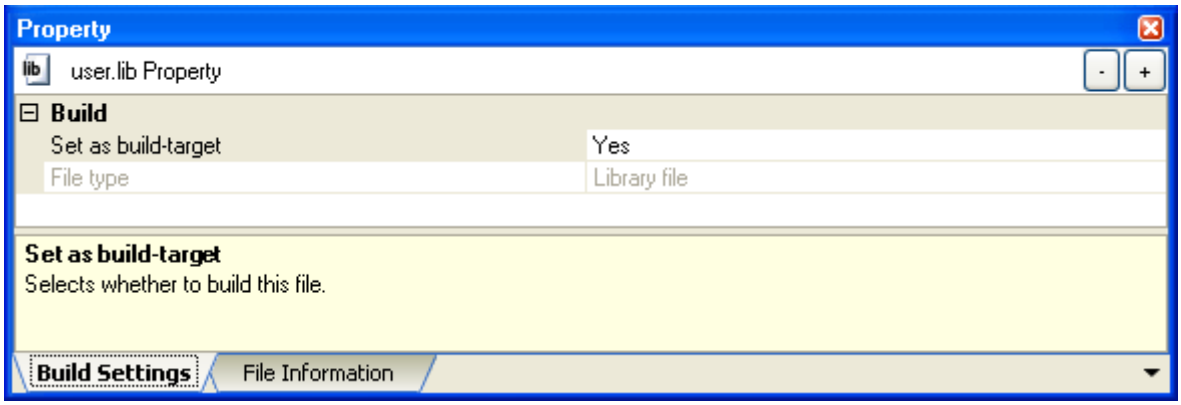


Figure A-13. Property Panel: [Build Settings] Tab (When Selecting Library File)



[Description of each category]

(1) [Build]

The detailed information on the build is displayed and the configuration can be changed.

Set as build-target	Select whether to run a build of the selected file.		
	Default	Yes	
	How to change	Select from the drop-down list.	
	Restriction	Yes	Runs a build of the selected file.
		No	Does not run a build of the selected file.
Set individual compile option	Select whether to set the compile option that differs from the project settings to the selected C source file. If [Yes(Level 3)(Perform with assuming it the whole program)(-Xwhole_program)] is selected in the [Perform inter-module optimization] property in the [Optimization(Details)] category from the [Compile Options] tab, this property will be grayed out and changed to [No]. This property is displayed only when a C source file is selected on the project tree and [Yes] in the [Set as build-target] property from this tab is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes	Sets the option that differs from the project settings to the selected C source file.
		No	Does not set the option that differs from the project settings to the selected C source file.

Set individual assemble option	Select whether to set the assemble option that differs from the project settings to the selected assembly source file. If [Yes(Level 3)(Perform with assuming it the whole program)(-Xwhole_program)] is selected in the [Perform inter-module optimization] property in the [Optimization(Details)] category from the [Compile Options] tab, this property will be grayed out and changed to [No]. This property is displayed only when the assembly source file is selected on the project tree and [Yes] in the [Set as build-target] property from this tab is selected.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes	Sets the option that differs from the project settings to the selected assembly source file.
		No	Does not set the option that differs from the project settings to the selected assembly source file.
File type	The type of the selected file is displayed.		
	Default	C source file (when the C source file is selected) Assembly source file (when the assembly source file is selected) Object file (when the object file is selected) Library file (when the library file is selected)	
	How to change	Changes not allowed	

[Individual Compile Options] tab

This tab shows the detailed information on a C source file categorized by the following and the configuration can be changed.

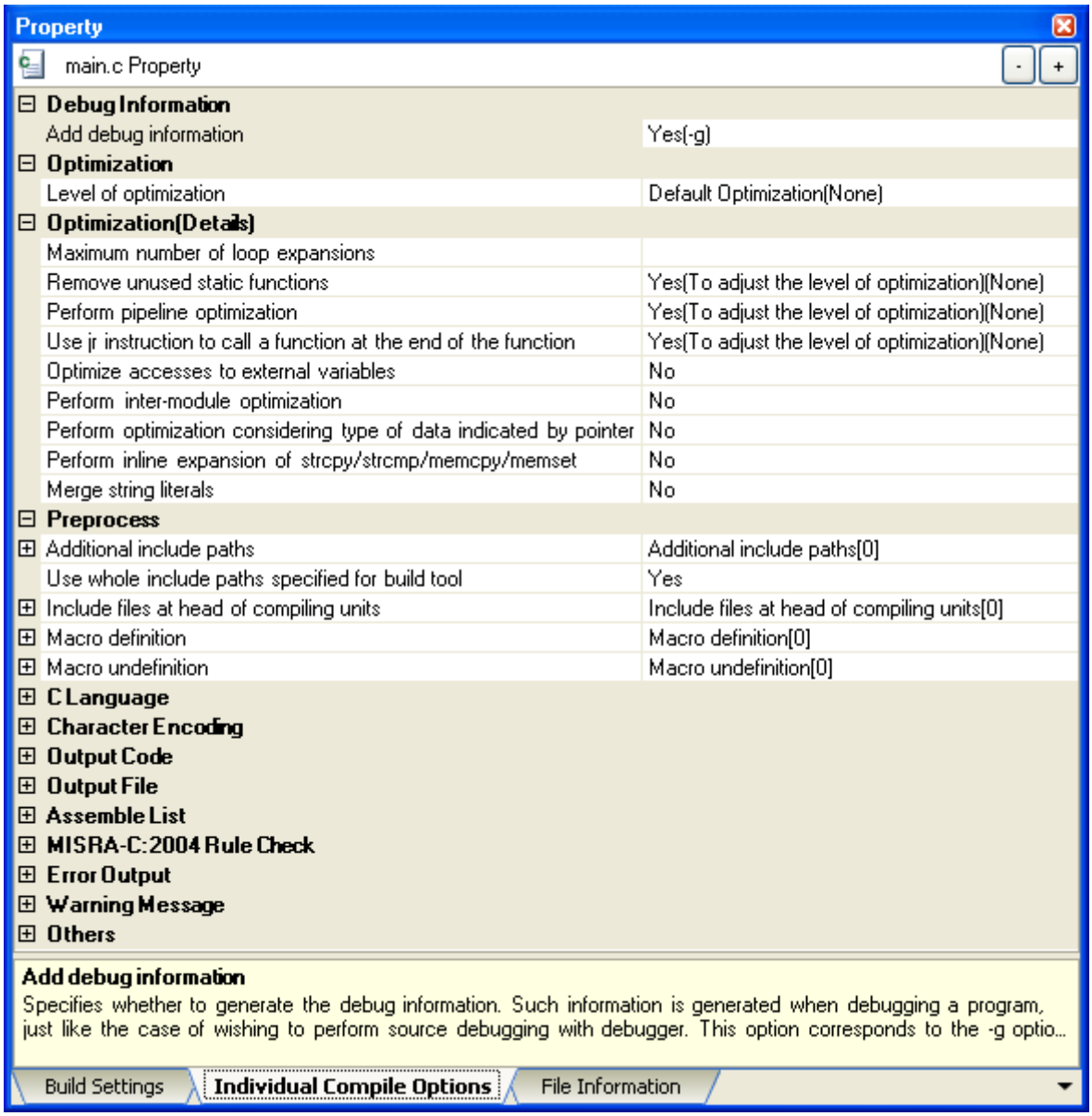
Note that this tab takes over the settings of the [Common Options] tab and [Compile Options] tab.

When the settings are changed from these tabs, the properties are displayed in boldface.

(1) [Debug Information]
(2) [Optimization]
(3) [Optimization(Details)]
(4) [Preprocess]
(5) [C Language]
(6) [Character Encoding]
(7) [Output Code]
(8) [Output File]
(9) [Assemble List]
(10) [MISRA-C:2004 Rule Check]
(11) [Error Output]
(12) [Warning Message]
(13) [Others]

Remark This tab is displayed only when [Yes] in the [Set individual compile option] property in the [Build] category from the [Build Settings] tab is selected.

Figure A-14. Property Panel: [Individual Compile Options] Tab



[Description of each category]

(1) [Debug Information]

The detailed information on debug information is displayed and the configuration can be changed.

Add debug information	Select whether to generate the debug information. It is possible to perform source debugging with the debugger by outputting information for source debugging to the output file. This corresponds to the -g option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-g)	Generates the debug information.
		No	Does not generate the debug information.

(2) [Optimization]

The detailed information on the optimization is displayed and the configuration can be changed.

Level of optimization	Select the level of the optimization for compiling. This corresponds to the -O option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Default Optimization(None)	Performs optimization that debugging is not affected (optimization of expressions and register allocation, and the like).
		Code Size Precedence(-Osize)	Performs optimization with the object size precedence. Regards reducing the ROM/RAM capacity as important and performs the maximum optimization that is effective for general programs.
		Speed Precedence(-Ospeed)	Performs optimization with the execution speed precedence. Regards shortening the execution speed as important and performs the maximum optimization that is effective for general programs.
		Debug Precedence(-Onothing)	Performs optimization with the debug precedence. Regards debugging as important and suppresses all optimization including default optimization.

(3) [Optimization(Details)]

The detailed information on the optimization is displayed and the configuration can be changed.

Maximum number of loop expansions	Specify the maximum number of times to expand the loops such as "for" and "while". If 0 or 1 is specified, expansion is suppressed. If this is blank, it is assumed that "4" has been specified. This corresponds to the -Ounroll option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Directly enter in the text box.	
	Restriction	0 to 999 (decimal number) or blank	
Remove unused static functions	Select whether to remove the static functions which are not called. This corresponds to the -Odelete_static_func option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(To adjust the level of optimization)(None)	Performs optimization according to the [Level of optimization] property.
		Yes(-Odelete_static_func)	Removes the unused static functions which are not called.
		No(-Odelete_static_func=off)	Does not remove the unused static functions which are not called.

<table>
<tr><td rowspan="8">Perform inline expansion</td><td colspan="3">Specify whether to perform inline expansion at the location calling functions.
This corresponds to the -Oinline option of the ccrh command.
This property is displayed only when [Code Size Precedence(-Osize)] or [Speed Precedence(-Ospeed)] in the [Level of optimization] property is selected.</td></tr>
<tr><td>Default</td><td colspan="2">Configuration of the compile option</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="5">Restriction</td><td>Yes(To adjust the level of optimization)(None)</td><td>Performs optimization according to the [Level of optimization] property.</td></tr>
<tr><td>Yes(Only specified functions)(-Oinline=1)</td><td>Performs inline expansion at the location calling the function for which #pragma inline is specified.</td></tr>
<tr><td>Yes(Auto-detect)(-Oinline=2)</td><td>Distinguishes the function that is the target of inline expansion automatically and expands it.</td></tr>
<tr><td>Yes(Auto-detect without code size increase)(-Oinline=3)</td><td>Distinguishes the function that is the target of inline expansion automatically and expands it, while minimizing the increase in code size.</td></tr>
<tr><td>No(-Oinline=0)</td><td>Suppresses all inline expansion including the function for which "#pragma inline" is specified.</td></tr>
<tr><td rowspan="4">Maximum increasing rate of inline expansion size</td><td colspan="3">Specify the maximum increasing rate (%) of the code size up to which inline expansion is performed. (Example: When "100" is specified, inline expansion will be applied until the code size increases by 100% (becomes twice the initial size).)
This corresponds to the -Oinline_size option of the ccrh command.
This property is displayed only when [Yes(Auto-detect)(-Oinline=2)] in the [Perform inline expansion] property is selected, or when [Yes(To adjust the level of optimization)] in the [Perform inline expansion] property and [Speed Precedence(-Ospeed)] in the [Optimization Level] property are selected.</td></tr>
<tr><td>Default</td><td colspan="2">Configuration of the compile option</td></tr>
<tr><td>How to change</td><td colspan="2">Directly enter in the text box.</td></tr>
<tr><td>Restriction</td><td colspan="2">0 to 65535 (decimal number)</td></tr>
<tr><td rowspan="5">Perform pipeline optimization</td><td colspan="3">Select whether to improve the program's execution performance by reordering instructions at the machine-language level.
This corresponds to the -Opipeline option of the ccrh command.</td></tr>
<tr><td>Default</td><td colspan="2">Configuration of the compile option</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="3">Restriction</td><td>Yes(To adjust the level of optimization)(None)</td><td>Performs optimization according to the [Level of optimization] property.</td></tr>
<tr><td>Yes(-Opipeline)</td><td>Performs pipeline optimization.</td></tr>
<tr><td>No(-Opipeline=off)</td><td>Does not perform pipeline optimization.</td></tr>
</table>

Use jr instruction to call a function at the end of the function	Select whether to give precedence to using jr instructions in the place of jarl instructions when the function ends with a function call. This corresponds to the -Otail_call option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(To adjust the level of optimization)(None)	Performs optimization according to the [Level of optimization] property.
		Yes(-Otail_call=on)	Gives precedence to using jr instructions in the place of jarl instructions when the function ends with a function call. The code size can be reduced by removing the store/restore instructions for lp. However, some debug functions cannot be used.
		No(-Otail_call=off)	Uses jarl instructions when the function ends with a function call.
Optimize accesses to external variables	Select whether to optimize accesses to external variables. This corresponds to the -Osmap and -Omap options of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(Optimizes the inner-module)(-Osmap)	Sets a base address for external or static variables defined in the file to be compiled, and generates code that accesses these relative to the base address.
		Yes(Optimizes the inter-module)(-Omap)	Generates an external symbol allocation information file. According to the information, recompilation is done to generate code that performs access to external or static variables relative to the base address. This item is displayed only when [No] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected. If [Yes] in the [Build simultaneously] property is selected when this item is selected, this item will be changed to [No].
		No	Does not optimize accesses to external variables.
Perform inter-module optimization	Specify the level of inter-module optimization (such as function merging). This corresponds to the -Xintermodule option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(Level 1)(Perform)(-Xintermodule)	Performs inter-module optimization for each file.
		No	Does not perform inter-module optimization.

Perform optimization considering type of data indicated by pointer	Select whether to perform optimization with consideration for the type of the data indicated by the pointer, based on the ANSI standard. This corresponds to the -Xalias option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xalias=ansi)	Performs optimization with consideration for the type of the data indicated by the pointer. In general, this option improves the object performance, but the execution result may differ from the case when [No] is selected.
		No	Does not perform optimization with consideration for the type of the data indicated by the pointer.
Perform inline expansion of strcpy/strcmp/ memcpy/memset	Selsect whether to perform inline expansion of functions "strcpy()", "strcmp()", "memcpy()", and "memset()" calls, with regarding the alignment conditions of the array (including character strings) and the structure as 4 bytes. This improves the execution speed of the program to be generated, but it increases the code size. This corresponds to the -Xinline_strcpy option of the ccrh command. This property is displayed only when [No] in the [Structure packing] property in the [Output Code] category from the [Compile Options] tab is selected.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xinline_strcpy)	Performs inline expansion of functions "strcpy()", "strcmp()", "memcpy()", and "memset()" calls.
		No	Does not perform inline expansion of functions "strcpy()", "strcmp()", "memcpy()", and "memset()" calls.
Merge string literals	When the same string literals exist in the source file, specify whether to merge them and allocate to the one area. This corresponds to the -Xmerge_string option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xmerge_string)	Merges the same string literals exist in the source file and allocates to the one area.
		No	Each allocates the same string literals exist in the source file to separate areas.

(4) [Preprocess]

The detailed information on preprocessing is displayed and the configuration can be changed.

<table>
<tr><td rowspan="4">Additional include paths</td><td colspan="3">Specify the additional include paths during compiling.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
The specified include path is searched with higher priority than the standard include file folder of CC-RH.
The reference point of the path is the project folder.
When this property is omitted, only the standard folder of CC-RH is searched.
This corresponds to the -I option of the ccrh command.
The specified include path is displayed as the subproperty.
Uppercase characters and lowercase characters are not distinguished for the include paths.</td></tr>
<tr><td>Default</td><td colspan="2">Additional include paths[number of defined items]</td></tr>
<tr><td>How to change</td><td colspan="2">Edit by the Path Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td colspan="2">Up to 259 characters
Up to 256 items can be specified.</td></tr>
<tr><td rowspan="4">Use whole include paths specified for build tool</td><td colspan="3">Select whether to compile using the include path specified in the [Additional include paths] property in the [Preprocess] category from the [Compile Options] tab of the build tool to be used.
The include paths are added by the following procedure.
- Paths specified in the [Additional include paths] property from this tab
- Paths specified in the [Additional include paths] property from the [Compile Options] tab
- Paths displayed in the [System include paths] property from the [Compile Options] tab
This corresponds to the -I option of the ccrh command.</td></tr>
<tr><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Restriction</td><td>Yes</td><td>Compiles using the include path specified in the property of the build tool to be used.</td></tr>
<tr><td>No</td><td>Does not use the include path specified in the property of the build tool to be used.</td></tr>
</table>

<table>
<tr><td rowspan="4">Include files at head of compiling units</td><td colspan="2">Specify the file that is included at the top of the compilation unit.
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
The reference point of the path is the project folder.
This corresponds to the -Xpreinclude option of the ccrh command.
The specified include file name is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Configuration of the compile option</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 247 characters
Up to 256 items can be specified.</td></tr>
<tr><td rowspan="4">Macro definition</td><td colspan="2">Specify the name of the macro to be defined.
Specify in the format of "macro name=defined value", with one macro name per line.
The "=defined value" part can be omitted, and in this case, "1" is used as the defined value.
This corresponds to the -D option of the ccrh command.
The specified macro is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Configuration of the compile option</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 256 characters
Up to 256 items can be specified.</td></tr>
<tr><td rowspan="4">Macro undefinition</td><td colspan="2">Specify the macro name to be undefined.
Specify in the format of "macro name", with one macro name per line.
This corresponds to the -U option of the ccrh command.
The specified macro is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>Configuration of the compile option</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 256 characters
Up to 256 items can be specified.</td></tr>
</table>

Output C source comments to preprocessed file	Select whether to output the comments of the C source to the preprocessed file. This corresponds to the -Xpreprocess option of the ccrh command. This property is displayed only when [Yes(-P)] in the [Output preprocessed source file] property in the [Output File] category is selected.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xpreprocess=comment)	Outputs the comments of the C source to the preprocessed file.
		No	Does not output the comments of the C source to the preprocessed file.
Output line number information to preprocessed file	Select whether to output the line number information of the C source to the preprocessed file. This corresponds to the -Xpreprocess option of the ccrh command. This property is displayed only when [Yes(-P)] in the [Output preprocessed source file] property in the [Output File] category is selected.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xpreprocess=line)	Outputs the line number information of the C source to the preprocessed file.
		No	Does not output the line number information of the C source to the preprocessed file.

(5) [C Language]

The detailed information on C language is displayed and the configuration can be changed.

Compile strictly according to ANSI standards	Select whether to process as making C source program comply strictly with the ANSI standard and output an error or warning for a specification that violates the standard. This corresponds to the -Xansi option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xansi)	Processes as making C source program comply strictly with the ANSI standard and outputs an error or warning for a specification that violates the standard.
		No	Compatibility with the conventional C language specifications is conferred and processing continues after warning is output.
Handle external variables as if they are volatile qualified	Select whether to handle all external variables as if they were volatile-declared. This corresponds to the -Xvolatile option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xvolatile)	Handles all external variables as if they were volatile-declared.
		No	Handles only the volatile-qualified variables as they were volatile-declared.

Check C program compatibility	Select whether to check the compatibility of a C program. This corresponds to the -Xcheck option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(for SuperH RISC engine C/C++ compiler)(-Xcheck=shc)	Checks the compatibility with the SuperH family C/C++ compiler.
		No	Does not check the compatibility with existing programs.

(6) [Character Encoding]

The detailed information on character encoding is displayed and the configuration can be changed.

Character encoding	Select the character code to be used for Japanese comments and character strings in the source file. This corresponds to the -Xcharacter_set option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Auto(None)	Interprets the Japanese character code in the source file as SJIS.
		SJIS(-Xcharacter_set=sjis)	Interprets the Japanese character code in the source file as SJIS.
		EUC(-Xcharacter_set=euc_jp)	Interprets the Japanese character code in the source file as EUC.
		UFT-8(-Xcharacter_set=utf8)	Interprets the Japanese character code in the source file as UFT-8.
		Big5(-Xcharacter_set=big5)	Interprets the Chinese character code in the source file as Traditional Chinese.
		GB2312(-Xcharacter_set=gb2312)	Interprets the Chinese character code in the source file as Simplified Chinese.
		No-process(-Xcharacter_set=none)	Does not interpret the Japanese/Chinese character code in the source file.

(7) [Output Code]

The detailed information on output codes is displayed and the configuration can be changed.

<table>
<tr><td rowspan="5">Output comment to assembly source file</td><td colspan="3">Select whether to output a C source program as a comment to the assembly source file to be output.
This corresponds to the -Xpass_source option of the ccrh command.
This property is displayed only when [Yes(-Xasm_path)] in the [Output assembly source file] property in the [Output File] category is selected or when [Yes(-Xasm_option=-Xprn_path)] in the [Output assemble list file] property in the [Assemble List] category is selected.</td></tr>
<tr><td>Default</td><td colspan="2">Configuration of the compile option</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Restriction</td><td>Yes(-Xpass_source)</td><td>Outputs a C source program as a comment to the assembly source file.</td></tr>
<tr><td>No</td><td>Does not output a C source program as a comment to the assembly source file.</td></tr>
<tr><td rowspan="6">Output code of switch statement</td><td colspan="3">Select the code output mode for switch statements in programs.
This corresponds to the -Xswitch option of the ccrh command.</td></tr>
<tr><td>Default</td><td colspan="2">Configuration of the compile option</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="3">Restriction</td><td>Auto(None)</td><td>The ccrh selects the optimum output format.</td></tr>
<tr><td>if-else(-Xswitch=ifelse)</td><td>Outputs the switch statements in the same format as the if-else statement along a string of case statements in programs.
Select this item if the case statements are written in the order of frequency or if only a few labels are used.
Because the case statements are compared starting from the top, unnecessary comparison can be reduced and the execution speed can be increased if the case statement that most often matches is written first.</td></tr>
<tr><td>Binary search(-Xswitch=binary)</td><td>Outputs the code in the binary search format for switch statements in programs.
Searches for a matching case statement by using a binary search algorithm.
If this item is selected when many labels are used, any case statement can be found at almost the same speed.</td></tr>
<tr><td></td><td>Table jump(-Xswitch=table)</td><td>Outputs the code in the table jump format for switch statements in programs.
References a table indexed on the values in the case statements, and selects and processes case labels from the switch statement values.
The code will branch to all the case statements with about the same speed.
However, if case values are not used in succession, an unnecessary area will be created.</td></tr>
</table>

Floating-point calculating type	Select whether to generate runtime library call instructions for floating-point calculations, or to generate floating-point instructions for the floating point unit (FPU). This corresponds to the -Xfloat option of the ccrh command. This property is displayed only when [Object for G3M(-Xcpu=g3m)] in the [Specify CPU core] property in the [Output File Type and Path] category from the [Common Options] tab is selected.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Auto(None)	Generates floating-point calculation instructions.
		Software Calculating(-Xfloat=soft)	Generates runtime library call instructions for floating-point calculations.
		FPU Calculating(-Xfloat=fpu)	Generates floating-point calculation instructions of FPU for floating-point calculations.
Generate div/divu instructions	Select whether to generate the div and divu instructions instead of the divq and divqu instructions for division. Although the divq and divqu instructions are fast, the number of execution cycles will differ depending on the values of the operands. This corresponds to the -Xdiv option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xdiv)	Generates the div and divu instructions for division.
		No	Generates the divq and divqu instructions for division.
Generate OV flag check code in division operation	Select whether to generate code (fetrap instruction) that checks the OV flag after division instructions and generate an FE level software exception when the OV flag is 1. This corresponds to the -Xcheck_div_ov option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xcheck_div_ov)	Generates code that checks the OV flag at division.
		No	Generates code that does not check the OV flag at division.
Vector number of fetrap instruction in divide exception	Specify the vector number of the fetrap instruction generated when the OV flag is 1. This corresponds to the -Xcheck_div_ov option of the ccrh command. This property is displayed only when [Yes(-Xcheck_div_ov)] in the [Generate OV flag check code in division operation] property is selected.		
	Default	*Configuration of the compile option*	
	How to change	Directly enter in the text box.	
	Restriction	1 to 15 (decimal number)	

Generate product-sum operation instruction	Select whether to generate product-sum operation instructions (fmaf.s, fmsf.s, fnmaf.s, and fnmsf.s) for single-precision floating-point product-sum operations. This corresponds to the -Xuse_fmaf option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xuse_fmaf)	Generates product-sum operation instructions for single-precision floating-point product-sum operations. Specifying this option will accelerate the execution speed but change the operation precision.
		No	Does not generate product-sum operation instructions.
Check invalid exception in cmpf instruction	Select whether to generate code by using the comparison condition for generating an invalid operation exception when any of the comparison values is a not-a-number in floating-point comparison. This corresponds to the -Xunordered_cmpf option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xunordered_cmpf)	Generates code by using the comparison condition for generating an invalid operation exception when any of the comparison values is a not-a-number in floating-point comparison.
		No	Does not detect invalid operation exceptions in floating-point comparison.
Use 32-bit branch instruction	Select whether to use the far jump function for the jarl and jr instructions. By using the far jump function, it is assumed that the jarl and jr instructions are jarl32 and jr32 instructions, and assembling is performed. This corresponds to the -Xasm_option=-Xasm_far_jump option of the ccrh command. This property is displayed only when [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xasm_option=-Xasm_far_jump)	Assumes that the jarl and jr instructions are jarl32 and jr32 instructions, and performs assembling.
		No	Performs assembly as a jarl or jr instruction.
Specify jump instruction	Specify the instruction to be generated for function-call branches. This corresponds to the -Xcall_jump option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Create jarl32 and jr32 instructions(-Xcall_jump=32)	Generates the jarl32 and jr32 instructions for the branch to the function.
		Create jarl and jr instructions(None)	Generates the jarl and jr instructions for the branch to the function.

Far Jump file names	Specify the Far Jump file name. The code that uses the jarl32 and jr32 instruction for branch instructions of functions described in a file is output to the Far Jump file. The ccrh command outputs an error if the function is in a range that cannot be branched to by the jarl or jr directive (±2MB or more), in which case the Far Jump file is used to recompile. Use the extension ".fjp". This corresponds to the -Xfar_jump option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Directly enter in the text box or edit by the Specify Far Jump File dialog box which appears when clicking the [...] button.	
	Restriction	Up to 259 characters	
Type of a generating program	Select the type of the program to be generated. This corresponds to the -Xmulti_level option of the ccrh command. This property is displayed only for the multi-core project.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Generate a program for single-core(None)	Generates a single-core program. The #pragma pmodule directives in the program are ignored.
		Generate a program for multi-core(-Xmulti_level=1)	Generates a multi-core program. The #pragma pmodule directives in the program become valid and the PM number is added to the end of the section name.

(8) [Output File]

The detailed information on output files is displayed and the configuration can be changed.

Object file name	Specify the name of the object file generated after compilation. The extension other than ".obj" cannot be specified. If the extension is omitted, ".obj" is automatically added. If this is blank, the file name will be the source file name with the extension replaced by ".obj". This corresponds to the -o option of the ccrh command.		
	Default	Blank	
	How to change	Directly enter in the text box.	
	Restriction	Up to 259 characters	
Output assembly source file	Select whether to output the assembly source file of the compile result for the C source. This corresponds to the -Xasm_path option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xasm_path)	Outputs the assembly source file of the compile result for the C source.
		No	Does not output the assembly source file of the compile result for the C source.

<table>
<tr><td rowspan="4">Output folder for assembly source file</td><td colspan="3">Specify the folder which the assembly source file is output.
If a relative path is specified, the reference point of the path is the main project or subproject folder.
If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different).
The following placeholder is supported.
%BuildModeName%: Replaces with the build mode name.
The assembly source file is saved under the C source file name with the extension replaced by ".asm".
If this is blank, it is assumed that the project folder has been specified.
This corresponds to the -Xasm_path option of the ccrh command.
This property is displayed only when [Yes(-Xasm_path)] in the [Output assembly source file] property is selected.</td></tr>
<tr><td>Default</td><td colspan="2">Configuration of the compile option</td></tr>
<tr><td>How to change</td><td colspan="2">Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td colspan="2">Up to 247 characters</td></tr>
<tr><td rowspan="4">Output preprocessed source file</td><td colspan="3">Select whether to output the execution result of preprocessing for the source file to a file.
This corresponds to the -P option of the ccrh command.</td></tr>
<tr><td>Default</td><td colspan="2">Configuration of the compile option</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Restriction</td><td>Yes(-P)</td><td>Outputs the execution result of preprocessing for the source file to a file.</td></tr>
<tr><td>No</td><td>Does not output the execution result of preprocessing for the source file to a file.</td></tr>
<tr><td rowspan="4">Output folder for preprocessed source file</td><td colspan="3">Specify the folder which the preprocessed source file is output.
The file is output under the source file name with the extension replaced by ".i".
If a relative path is specified, the reference point of the path is the main project or subproject folder.
If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different).
The following placeholder is supported.
%BuildModeName%: Replaces with the build mode name.
If this is blank, it is assumed that the project folder has been specified.
This corresponds to the -Xprep_path option of the ccrh command.
This property is displayed only when [Yes(-P)] in the [Output preprocessed source file] property is selected.</td></tr>
<tr><td>Default</td><td colspan="2">Configuration of the compile option</td></tr>
<tr><td>How to change</td><td colspan="2">Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td colspan="2">Up to 247 characters</td></tr>
</table>

(9) [Assemble List]

The detailed information on the assemble list is displayed and the configuration can be changed.

Output assemble list file	Select whether to output the assemble list file. This corresponds to the -Xasm_option=-Xprn_path option of the ccrh command.		
	Default	*Configuration of the compile option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xasm_option=-Xprn_path)	Outputs the assemble list file.
		No	Does not output the assemble list file.
Output folder for assemble list file	Specify the folder which the assemble list file is output. The assemble list file is output under the source file name with the extension replaced by ".prn". If a relative path is specified, the reference point of the path is the main project or subproject folder. If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different). The following placeholder is supported. %BuildModeName%: Replaces with the build mode name. If this is blank, it is assumed that the project folder has been specified. This corresponds to the -Xasm_option=-Xprn_path option of the ccrh command. This property is displayed only when [Yes(-Xasm_option=-Xprn_path)] in the [Output assemble list file] property is selected.		
	Default	*Configuration of the compile option*	
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.	
	Restriction	Up to 247 characters	

(10) [MISRA-C:2004 Rule Check]

The detailed information on the MISRA-C:2004 rule check are displayed and the configuration can be changed.

<table>
<tr><td rowspan="11">Apply rule</td><td colspan="3">Select the MISRA-C:2004 rules to be applied.
This corresponds to the -Xmisra2004 option of the ccrh command.</td></tr>
<tr><td>Default</td><td colspan="2">Configuration of the compile option</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="8">Restriction</td><td>Apply all rules(-Xmisra2004=all)</td><td>Checks the source code against all of the rules which are supported.</td></tr>
<tr><td>Apply specified rule number(-Xmisra2004=apply)</td><td>Checks the source code against the rules with the specified numbers among the rules which are supported.</td></tr>
<tr><td>Ignore specified rule number(-Xmisra2004=ignore)</td><td>Checks the source code against the rules that do not match the specified numbers among the rules which are supported.</td></tr>
<tr><td>Apply rules that are classified as "required"(-Xmisra2004=required)</td><td>Checks the source code against the rules of the "required" type.</td></tr>
<tr><td>Apply rules that are classified as "required" and specified rule number(-Xmisra2004=required_add)</td><td>Checks the source code against the rules of the "required" type and the rules with the specified numbers among the rules which are supported.</td></tr>
<tr><td>Ignore specified rule number from rules that are classified as "required"(-Xmisra2004=required_remove)</td><td>Checks the source code against the rules of the "required" type except for the rules with the specified numbers among the rules which are supported.</td></tr>
<tr><td>Apply rules that are described in the specified file(-Xmisra2004=<file name>)</td><td>Checks the source code against the rules with the numbers described in specified file among the rules which are supported.</td></tr>
<tr><td>Not apply rule(None)</td><td>Does not apply the MISRA-C:2004 rules.</td></tr>
<tr><td rowspan="4">Rule number description file</td><td colspan="3">Specify the rule number description file (MISRA-C:2004 rule file).
The following placeholders are supported.
%BuildModeName%: Replaces with the build mode name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectName%: Replaces with the project name.
This corresponds to the -Xmisra2004 option of the ccrh command.
This property is displayed only when [Apply rules that are described in the specified file(-Xmisra2004=<file name>)] in the [Apply rule] property is selected.</td></tr>
<tr><td>Default</td><td colspan="2">Configuration of the compile option</td></tr>
<tr><td>How to change</td><td colspan="2">Directly enter in the text box or edit by the Specify MISRA-C:2004 Rule File dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td colspan="2">Up to 259 characters</td></tr>
</table>

Rule number	Specify the rule number to be checked. Specify at least one rule number in decimal. This corresponds to the -Xmisra2004 option of the ccrh command. This property is displayed only when [Apply specified rule number(-Xmisra2004=apply)] in the [Apply rule] property is selected.	
	Default	*Configuration of the compile option*
	How to change	Directly enter in the text box or edit by the Specify Rule Number dialog box which appears when clicking the [...] button.
	Restriction	Up to 259 characters
Exclusion rule number	Specify the rule number to be excluded from the check. Specify at least one rule number in decimal. This corresponds to the -Xmisra2004 option of the ccrh command. This property is displayed only when [Ignore specified rule number(-Xmisra2004=ignore)] in the [Apply rule] property is selected.	
	Default	*Configuration of the compile option*
	How to change	Directly enter in the text box or edit by the Specify Rule Number dialog box which appears when clicking the [...] button.
	Restriction	Up to 259 characters
Check rule number besides required rule	Specify the rule number to be checked besides the required rules. Specify at least one rule number in decimal. This corresponds to the -Xmisra2004 option of the ccrh command. This property is displayed only when [Apply rules that are classified as "required" and specified rule number(-Xmisra2004=required_add)] in the [Apply rule] property is selected.	
	Default	*Configuration of the compile option*
	How to change	Directly enter in the text box or edit by the Specify Rule Number dialog box which appears when clicking the [...] button.
	Restriction	Up to 259 characters
Exclusion rule number from required rule	Specify the required rule number to be excluded from the check. Specify at least one rule number in decimal. This corresponds to the -Xmisra2004 option of the ccrh command. This property is displayed only when [Ignore specified rule number from rules that are classified as "required"(-Xmisra2004=required_remove)] in the [Apply rule] property is selected.	
	Default	*Configuration of the compile option*
	How to change	Directly enter in the text box or edit by the Specify Rule Number dialog box which appears when clicking the [...] button.
	Restriction	Up to 259 characters

<table>
<tr><td rowspan="4">Rule check exclusion file</td><td colspan="2">Specify files that will not be checked against the MISRA-C:2004 rules.
The following placeholders are supported.
%BuildModeName%: Replaces with the build mode name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%ProjectName%: Replaces with the project name.
This corresponds to the -Xignore_files_misra option of the ccrh command.
This property is displayed only in the following cases.
- When [Apply all rules] is selected in the [Apply rule] property
- When [Apply rules that are classified as "required"] is selected in the [Apply rule] property
- When [Apply specified rule number] is selected in the [Apply rule] property and a rule number is specified in the [Rule number] property
- When [Ignore specified rule number] is selected in the [Apply rule] property and a rule number is specified in the [Rule number] property
- When [Apply rules that are classified as "required" and specified rule number] is selected in the [Apply rule] property and a rule number is specified in the [Check rule number besides required rule] property
- When [Ignore specified rule number from rules that are classified as "required"] is selected in the [Apply rule] property and a rule number is specified in the [Exclusion rule number from required rule] property
- When [Apply rules that are described in the specified file] is selected in the [Apply rule] property and a rule number description file is specified in the [Rule number description file] property</td></tr>
<tr><td>Default</td><td>Configuration of the compile option</td></tr>
<tr><td>How to change</td><td>Edit by the Path Edit dialog box which appears when clicking the [...] button.
-> Edit by the Add Excluding File dialog box which appears when clicking the [Browse...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 259 characters</td></tr>
</table>

<table>
<tr><td rowspan="5">Output message of the enhanced key word and extended specifications</td><td colspan="3">Select whether to output the message of the enhanced key word and extended specifications.
This corresponds to the -Xcheck_language_extention option of the ccrh command.
This property is displayed only in the following cases.
- When [Apply all rules] is selected in the [Apply rule] property
- When [Apply rules that are classified as "required"] is selected in the [Apply rule] property
- When [Apply specified rule number] is selected in the [Apply rule] property and a rule number is specified in the [Rule number] property
- When [Ignore specified rule number] is selected in the [Apply rule] property and a rule number is specified in the [Rule number] property
- When [Apply rules that are classified as "required" and specified rule number] is selected in the [Apply rule] property and a rule number is specified in the [Check rule number besides required rule] property
- When [Ignore specified rule number from rules that are classified as "required"] is selected in the [Apply rule] property and a rule number is specified in the [Exclusion rule number from required rule] property
- When [Apply rules that are described in the specified file] is selected in the [Apply rule] property and a rule number description file is specified in the [Rule number description file] property</td></tr>
<tr><td>Default</td><td colspan="2">Configuration of the compile option</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Restriction</td><td>Yes(-Xcheck_language_extension)</td><td>Enables MISRA-C:2004 rule check and outputs messages when the rule check is partially suppressed by the unique language specifications extended from the C language standard.</td></tr>
<tr><td>No</td><td>Disables MISRA-C:2004 rule check is disabled, which are partially suppressed by the extended language specifications.</td></tr>
</table>

(11) [Error Output]

The detailed information on the error output is displayed and the configuration can be changed.

<table>
<tr><td rowspan="4">Output error message file</td><td colspan="3">Select whether to output the error message file.
This corresponds to the -Xerror_file option of the ccrh command.
Error messages are displayed on the Output panel regardless of this property's .</td></tr>
<tr><td>Default</td><td colspan="2">Configuration of the common option</td></tr>
<tr><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td rowspan="2">Restriction</td><td>Yes(-Xerror_file)</td><td>Outputs the error message file.</td></tr>
<tr><td>No</td><td>Does not output the error message file.</td></tr>
</table>

Error message file output folder	Specify the folder which the error message file is output. If a relative path is specified, the reference point of the path is the main project or subproject folder. If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different). The following placeholder is supported. %BuildModeName%: Replaces with the build mode name. If this is blank, it is assumed that the project folder has been specified. This corresponds to the -Xerror_file option of the ccrh command. This property is displayed only when [Yes(-Xerror_file)] in the [Output error message file] property is selected.	
	Default	*Configuration of the common option*
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.
	Restriction	Up to 247 characters
Error message file name	Specify the error message file name. The extension can be freely specified. The following placeholders are supported. %ActiveProjectName%: Replaces with the active project name. %MainProjectName%: Replaces with the main project name. %ProjectName%: Replaces with the project name. If this is blank, it is assumed that "%ProjectName%.err" has been specified. This corresponds to the -Xerror_file option of the ccrh command. This property is displayed only when [Yes(-Xerror_file)] in the [Output error message file] property is selected.	
	Default	*Configuration of the common option*
	How to change	Directly enter in the text box.
	Restriction	Up to 259 characters

(12) [Warning Message]

The detailed information on warning messages is displayed and the configuration can be changed.

Undisplayed warning message	Specify the number of the warning message not to be displayed. If multiple message numbers are specified, delimit them with "," (comma) (example: 02042,02107). Also, the range can be set using "-" (hyphen) (example: 02222-02554,02699-02782). This corresponds to the -Xno_warning option of the ccrh command.	
	Default	*Configuration of the common option*
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.
	Restriction	Up to 2048 characters

(13) [Others]

Other detailed information on compilation is displayed and the configuration can be changed.

<table>
<tr><td>Commands executed before compile processing</td><td colspan="2">Specify the command to be executed before compile processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%CompiledFile%: Replaces with the absolute path of the output file under compiling.
%InputFile%: Replaces with the absolute path of the file to be compiled.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%Options%: Replaces with the command line option under build execution.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%Program%: Replaces with the program name under execution.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed before compile processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected
Blank
- Other than above
Configuration of the compile option</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr>
</table>

<table>
<tr><td rowspan="4">Commands executed after compile processing</td><td colspan="2">Specify the command to be executed after compile processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%BuildModeName%: Replaces with the build mode name.
%CompiledFile%: Replaces with the absolute path of the output file under compiling.
%InputFile%: Replaces with the absolute path of the file to be compiled.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%Options%: Replaces with the command line option under build execution.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%Program%: Replaces with the program name under execution.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed after compile processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected
Blank
- Other than above
Configuration of the compile option</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr>
<tr><td rowspan="4">Other additional options</td><td colspan="2">Input the compile option to be added additionally.
The options set here are added at the end of the compile options group.</td></tr>
<tr><td>Default</td><td>- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected
Blank
- Other than above
Configuration of the compile option</td></tr>
<tr><td>How to change</td><td>Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td>Up to 259 characters</td></tr>
</table>

[Individual Assemble Options] tab

This tab shows the detailed information on an assemble source file categorized by the following and the configuration can be changed.

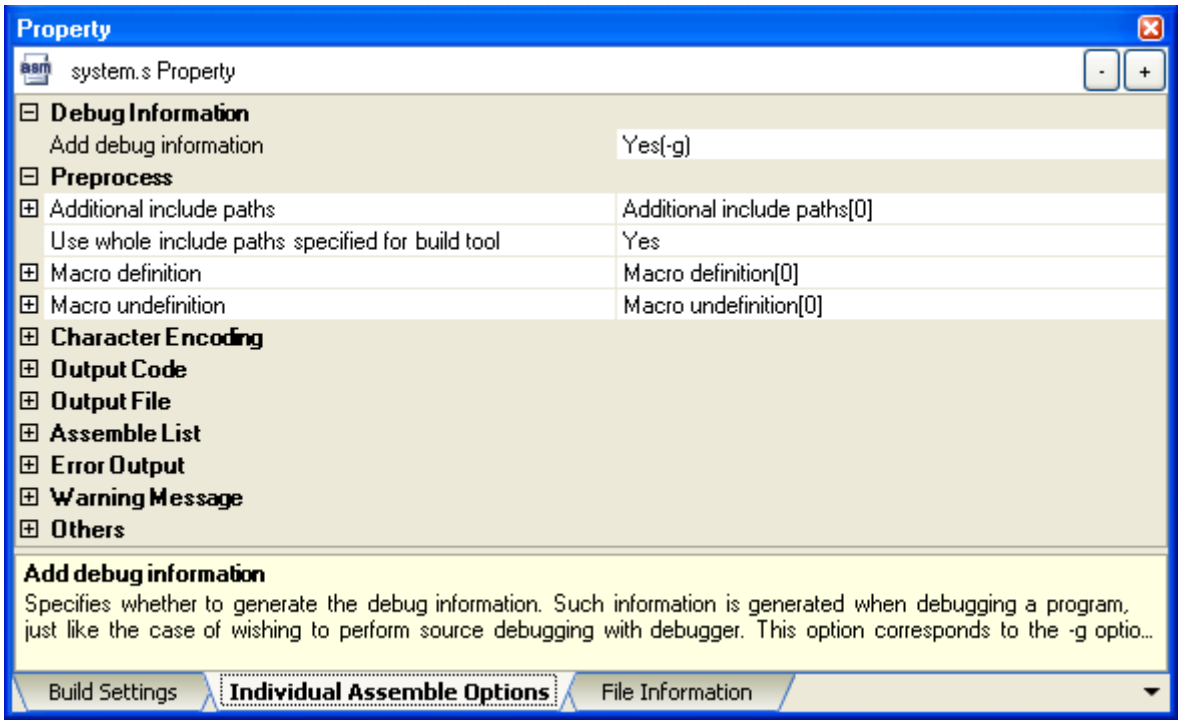
Note that this tab takes over the settings of the [Common Options] tab, [Compile Options] tab, and [Assemble Options] tab.

When the settings are changed from these tabs, the properties are displayed in boldface.

(1) [Debug Information]
(2) [Preprocess]
(3) [Character Encoding]
(4) [Output Code]
(5) [Output File]
(6) [Assemble List]
(7) [Error Output]
(8) [Warning Message]
(9) [Others]

Remark This tab is displayed only when [Yes] in the [Set individual assemble option] property in the [Build] category from the [Build Settings] tab is selected.

Figure A-15. Property Panel: [Individual Assemble Options] Tab



[Description of each category]

(1) [Debug Information]

The detailed information on debug information is displayed and the configuration can be changed.

Add debug information	Select whether to generate the debug information. It is possible to perform source debugging with the debugger by outputting information for source debugging to the output file. This corresponds to the -g option of the ccrh command.		
	Default	- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected *Configuration of the compile option* - Other than above *Configuration of the assemble option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-g)	Generates the debug information.
		No	Does not generate the debug information.

(2) [Preprocess]

The detailed information on preprocessing is displayed and the configuration can be changed.

Additional include paths	Specify the additional include paths during assembling. The following placeholders are supported. %ActiveProjectDir%: Replaces with the absolute path of the active project folder. %ActiveProjectName%: Replaces with the active project name. %BuildModeName%: Replaces with the build mode name. %MainProjectDir%: Replaces with the absolute path of the main project folder. %MainProjectName%: Replaces with the main project name. %MicomToolPath%: Replaces with the absolute path of the install folder of this product. %ProjectDir%: Replaces with the absolute path of the project folder. %ProjectName%: Replaces with the project name. %TempDir%: Replaces with the absolute path of the temporary folder. %WinDir%: Replaces with the absolute path of the Windows system folder. The specified include path is searched with higher priority than the standard include file folder of CC-RH. The reference point of the path is the project folder. When this property is omitted, only the standard folder of CC-RH is searched. This corresponds to the -I option of the ccrh command. The specified include path is displayed as the subproperty. Uppercase characters and lowercase characters are not distinguished for the include paths.	
	Default	Additional include paths[*number of defined items*]	
	How to change	Edit by the Path Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.	
	Restriction	Up to 259 characters Up to 256 items can be specified.	

<table>
<tr><td>Use whole include paths specified for build tool</td><td colspan="3">Select whether to assemble using the include path specified in the [Additional include paths] property in the [Preprocess] category from the [Assemble Options] tab of the build tool to be used.
The setting of the [Compile Options] tab is used when [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected.
When [Yes] in the [Build simultaneously] property is selected, the include paths are added by the following procedure.
- Paths specified in the [Additional include paths] property from this tab
- Paths specified in the [Additional include paths] property from the [Compile Options] tab
- Paths displayed in the [System include paths] property from the [Compile Options] tab
When [No] in the [Build simultaneously] property is selected, the include paths are added by the following procedure.
- Paths specified in the [Additional include paths] property from this tab
- Paths specified in the [Additional include paths] property from the [Assemble Options] tab
- Paths displayed in the [System include paths] property from the [Assemble Options] tab
This corresponds to the -I option of the ccrh command.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">Yes</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Select from the drop-down list.</td></tr>
<tr><td></td><td rowspan="2">Restriction</td><td>Yes</td><td>Assembles using the include path specified in the property of the build tool to be used.</td></tr>
<tr><td></td><td>No</td><td>Does not use the include path specified in the property of the build tool to be used.</td></tr>
<tr><td>Macro definition</td><td colspan="3">Specify the name of the macro to be defined.
Specify in the format of "macro name=defined value", with one macro name per line.
The "=defined value" part can be omitted, and in this case, "1" is used as the defined value.
This corresponds to the -D option of the ccrh command.
The specified macro is displayed as the subproperty.</td></tr>
<tr><td></td><td>Default</td><td colspan="2">- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected
Configuration of the compile option
- Other than above
Configuration of the assemble option</td></tr>
<tr><td></td><td>How to change</td><td colspan="2">Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td></td><td>Restriction</td><td colspan="2">Up to 256 characters
Up to 256 items can be specified.</td></tr>
</table>

Macro undefinition	Specify the macro name to be undefined. Specify in the format of "*macro name*", with one macro name per line. This corresponds to the -U option of the ccrh command. The specified macro is displayed as the subproperty.	
	Default	- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected *Configuration of the compile option* - Other than above *Configuration of the assemble option*
	How to change	Edit by the Text Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.
	Restriction	Up to 256 characters Up to 256 items can be specified.

(3) [Character Encoding]

The detailed information on character encoding is displayed and the configuration can be changed.

Character encoding	Select the character code to be used for Japanese comments and character strings in the source file. This corresponds to the -Xcharacter_set option of the ccrh command.		
	Default	- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected *Configuration of the compile option* - Other than above *Configuration of the assemble option*	
	How to change	Select from the drop-down list.	
	Restriction	Auto(None)	Interprets the Japanese character code in the source file as SJIS.
		SJIS(-Xcharacter_set=sjis)	Interprets the Japanese character code in the source file as SJIS.
		EUC(-Xcharacter_set=euc_jp)	Interprets the Japanese character code in the source file as EUC.
		UFT-8(-Xcharacter_set=utf8)	Interprets the Japanese character code in the source file as UFT-8.
		Big5(-Xcharacter_set=big5)	Interprets the Chinese character code in the source file as Traditional Chinese.
		GB2312(-Xcharacter_set=gb2312)	Interprets the Chinese character code in the source file as Simplified Chinese.
		No-process(-Xcharacter_set=none)	Does not interpret the Japanese/Chinese character code in the source file.

(4) [Output Code]

The detailed information on output codes is displayed and the configuration can be changed.

Use 32-bit branch instruction	Select whether to use the far jump function for the jarl and jr instructions. By using the far jump function, it is assumed that the jarl and jr instructions are jarl32 and jr32 instructions, and assembling is performed. This corresponds to the -Xasm_option=-Xasm_far_jump option of the ccrh command.		
	Default	- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected *Configuration of the compile option* - Other than above *Configuration of the assemble option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xasm_option=-Xasm_far_jump)	Assumes that the jarl and jr instructions are jarl32 and jr32 instructions, and performs assembling.
		No	Performs assembly as a jarl or jr instruction.

(5) [Output File]

The detailed information on output files is displayed and the configuration can be changed.

Object file name	Specify the name of the object file generated after assembling. The extension other than ".obj" cannot be specified. If the extension is omitted, ".obj" is automatically added. If this is blank, the file name will be the source file name with the extension replaced by ".obj". This corresponds to the -o option of the ccrh command.	
	Default	Blank
	How to change	Directly enter in the text box.
	Restriction	Up to 259 characters

(6) [Assemble List]

The detailed information on the assemble list is displayed and the configuration can be changed.

Output assemble list file	Select whether to output the assemble list file. This corresponds to the -Xasm_option=-Xprn_path option of the ccrh command.		
	Default	- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected *Configuration of the compile option* - Other than above *Configuration of the assemble option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xasm_option=-Xprn_path)	Outputs the assemble list file.
		No	Does not output the assemble list file.

Output folder for assemble list file	Specify the folder which the assemble list file is output. The assemble list file is output under the source file name with the extension replaced by ".prn". If a relative path is specified, the reference point of the path is the main project or subproject folder. If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different). The following placeholder is supported. %BuildModeName%: Replaces with the build mode name. If this is blank, it is assumed that the project folder has been specified. This corresponds to the -Xasm_option=-Xprn_path option of the ccrh command. This property is displayed only when [Yes(-Xasm_option=-Xprn_path)] in the [Output assemble list file] property is selected.	
	Default	- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected *Configuration of the compile option* - Other than above *Configuration of the assemble option*
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.
	Restriction	Up to 247 characters

(7) [Error Output]

The detailed information on the error output is displayed and the configuration can be changed.

Output error message file	Select whether to output the error message file. This corresponds to the -Xerror_file option of the ccrh command. Error messages are displayed on the Output panel regardless of this property's.		
	Default	*Configuration of the common option*	
	How to change	Select from the drop-down list.	
	Restriction	Yes(-Xerror_file)	Outputs the error message file.
		No	Does not output the error message file.
Error message file output folder	Specify the folder which the error message file is output. If a relative path is specified, the reference point of the path is the main project or subproject folder. If an absolute path is specified, the reference point of the path is the main project or subproject folder (unless the drives are different). The following placeholder is supported. %BuildModeName%: Replaces with the build mode name. If this is blank, it is assumed that the project folder has been specified. This corresponds to the -Xerror_file option of the ccrh command. This property is displayed only when [Yes(-Xerror_file)] in the [Output error message file] property is selected.		
	Default	*Configuration of the common option*	
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.	
	Restriction	Up to 247 characters	

Error message file name	Specify the error message file name. The extension can be freely specified. The following placeholders are supported. %ActiveProjectName%: Replaces with the active project name. %MainProjectName%: Replaces with the main project name. %ProjectName%: Replaces with the project name. If this is blank, it is assumed that "%ProjectName%.err" has been specified. This corresponds to the -Xerror_file option of the ccrh command. This property is displayed only when [Yes(-Xerror_file)] in the [Output error message file] property is selected.	
	Default	*Configuration of the common option*
	How to change	Directly enter in the text box.
	Restriction	Up to 259 characters

(8) [Warning Message]

The detailed information on warning messages is displayed and the configuration can be changed.

Undisplayed warning message	Specify the number of the warning message not to be displayed. If multiple message numbers are specified, delimit them with "," (comma) (example: 02042,02107). Also, the range can be set using "-" (hyphen) (example: 02222-02554,02699-02782). This corresponds to the -Xno_warning option of the ccrh command.	
	Default	*Configuration of the common option*
	How to change	Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.
	Restriction	Up to 2048 characters

(9) [Others]

Other detailed information on assembly is displayed and the configuration can be changed.

<table>
<tr><td rowspan="4">Commands executed before assemble processing</td><td colspan="2">Specify the command to be executed before assemble processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%AssembledFile%: Replaces with the absolute path of the output file under assembling.
%BuildModeName%: Replaces with the build mode name.
%InputFile%: Replaces with the absolute path of the file to be assembled.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%Options%: Replaces with the command line option under build execution.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%Program%: Replaces with the program name under execution.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed before assemble processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected
Blank
- Other than above
Configuration of the assemble option</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr>
</table>

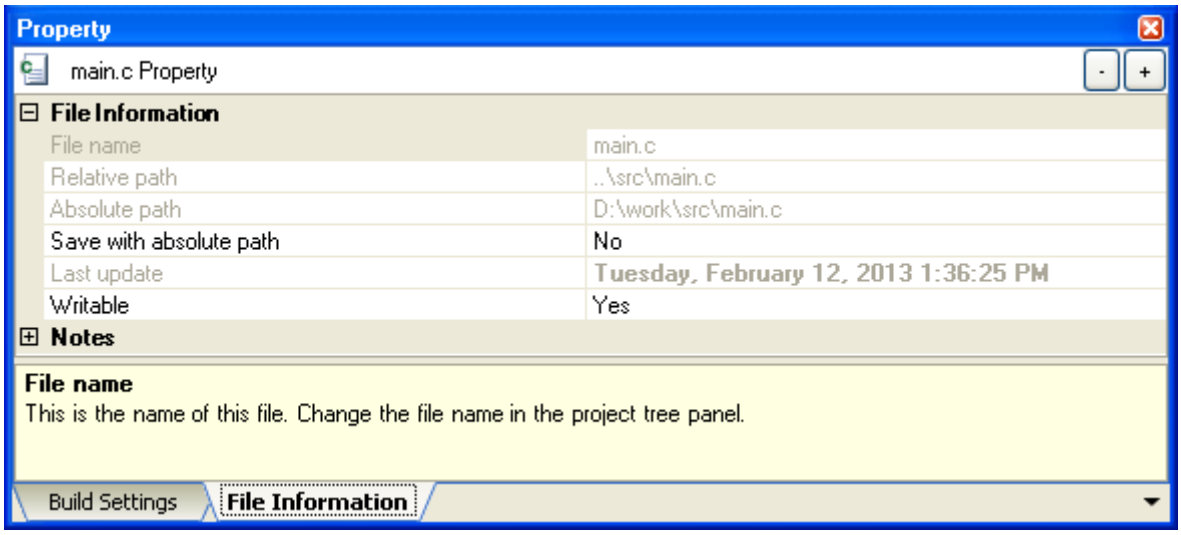
<table>
<tr><td rowspan="4">Commands executed after assemble processing</td><td colspan="2">Specify the command to be executed after assemble processing.
Use the call instruction to specify a batch file (example: call a.bat).
The following placeholders are supported.
%ActiveProjectDir%: Replaces with the absolute path of the active project folder.
%ActiveProjectName%: Replaces with the active project name.
%AssembledFile%: Replaces with the absolute path of the output file under assembling.
%BuildModeName%: Replaces with the build mode name.
%InputFile%: Replaces with the absolute path of the file to be assembled.
%MainProjectDir%: Replaces with the absolute path of the main project folder.
%MainProjectName%: Replaces with the main project name.
%MicomToolPath%: Replaces with the absolute path of the install folder of this product.
%Options%: Replaces with the command line option under build execution.
%OutputDir%: Replaces with the absolute path of the output folder.
%OutputFile%: Replaces with the absolute path of the output file.
%Program%: Replaces with the program name under execution.
%ProjectDir%: Replaces with the absolute path of the project folder.
%ProjectName%: Replaces with the project name.
%TempDir%: Replaces with the absolute path of the temporary folder.
%WinDir%: Replaces with the absolute path of the Windows system folder.
When "#!python" is described in the first line, the contents from the second line to the last line are regarded as the script of the Python console, and then executed after assemble processing.
The placeholders can be described in the script.
The specified command is displayed as the subproperty.</td></tr>
<tr><td>Default</td><td>- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected
Blank
- Other than above
Configuration of the assemble option</td></tr>
<tr><td>How to change</td><td>Edit by the Text Edit dialog box which appears when clicking the [...] button.
For the subproperty, you can enter directly in the text box.</td></tr>
<tr><td>Restriction</td><td>Up to 1023 characters
Up to 64 items can be specified.</td></tr>
<tr><td rowspan="4">Other additional options</td><td colspan="2">Input the assemble option to be added additionally.
The options set here are added at the end of the assemble options group.</td></tr>
<tr><td>Default</td><td>- When [Yes] in the [Build simultaneously] property in the [Build Method] category from the [Common Options] tab is selected
Blank
- Other than above
Configuration of the assemble option</td></tr>
<tr><td>How to change</td><td>Directly enter in the text box or edit by the Character String Input dialog box which appears when clicking the [...] button.</td></tr>
<tr><td>Restriction</td><td>Up to 259 characters</td></tr>
</table>

[File Information] tab

This tab shows the detailed information on each file categorized by the following and the configuration can be changed.

(1) [File Information]
(2) [Notes]

Figure A-16. Property Panel: [File Information] Tab



[Description of each category]

(1) [File Information]

The detailed information on the file is displayed and the configuration can be changed.

File name	The file name is displayed. Change the file name on the project tree.	
	Default	*file name*
	How to change	Changes not allowed
Relative path	The relative path from the project folder is displayed.	
	Default	*The relative path from the project folder*
	How to change	Changes not allowed
Absolute path	The absolute path of the file is displayed.	
	Default	*The absolute path of the file*
	How to change	Changes not allowed

Save with absolute path	Select whether to save the file location with the absolute path. This property is not displayed if the Property panel being displayed while a dependency file is selected on the project tree.		
	Default	No	
	How to change	Select from the drop-down list.	
	Restriction	Yes	Saves the file location with the absolute path.
		No	Saves the file location with the relative path.
Last update	The time and date on which this file was changed last is displayed.		
	Default	*File updated time and date*	
	How to change	Changes not allowed	
Writable	Select whether to enable writing to the file. This property is not displayed if the Property panel being displayed while a dependency file is selected on the project tree.		
	Default	Yes (when the file is write enabled) No (when the file is not write enabled)	
	How to change	Select from the drop-down list.	
	Restriction	Yes	Enables the file to write.
		No	Does not enable the file to write.

(2) [Notes]

The detailed information on notes is displayed and the configuration can be changed.

This category is not displayed if the Property panel being displayed while a dependency file is selected on the project tree.

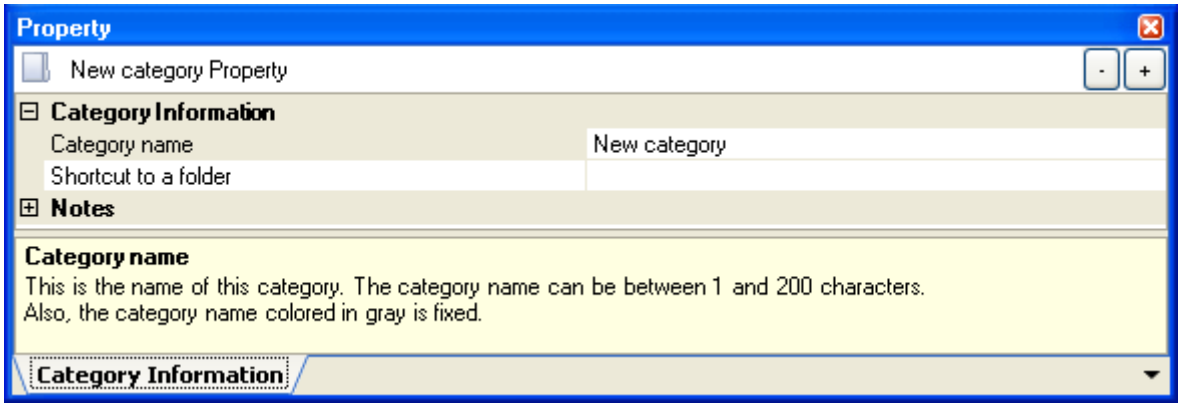
Memo	Add memos to the file. Add one item in one line. The specified memo is displayed as the subproperty.	
	Default	Memo[*number of items*]
	How to change	Edit by the Text Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.
	Restriction	Up to 256 characters Up to 256 items can be specified.

[Category Information] tab

This tab shows the detailed information on the category node (the category that the user added), Files node, and Build tool generated files node categorized by the following and the configuration can be changed.

(1) [Category Information]
(2) [Notes]

Figure A-17. Property Panel: [Category Information] Tab



[Description of each category]

(1) [Category Information]

The detailed information on the category is displayed and the configuration can be changed.

Category name	Specify the name of the category to categorize files. This property of the Files node and Build tool generated files node is displayed in gray and you cannot change the attribute.	
	Default	*Category name of files*
	How to change	Directly enter in the text box.
	Restriction	1 to 200 characters
Shortcut to a folder	Specify a shortcut to a folder. If a relative path is specified, the reference point of the path is the main project or subproject folder. This property of the Files node and Build tool generated files node is not displayed.	
	Default	Blank
	How to change	Directly enter in the text box or edit by the Browse For Folder dialog box which appears when clicking the [...] button.
	Restriction	Up to 247 characters

(2) [Notes]

The detailed information on notes is displayed and the configuration can be changed.
This category of the Files node and Build tool generated files node is not displayed.

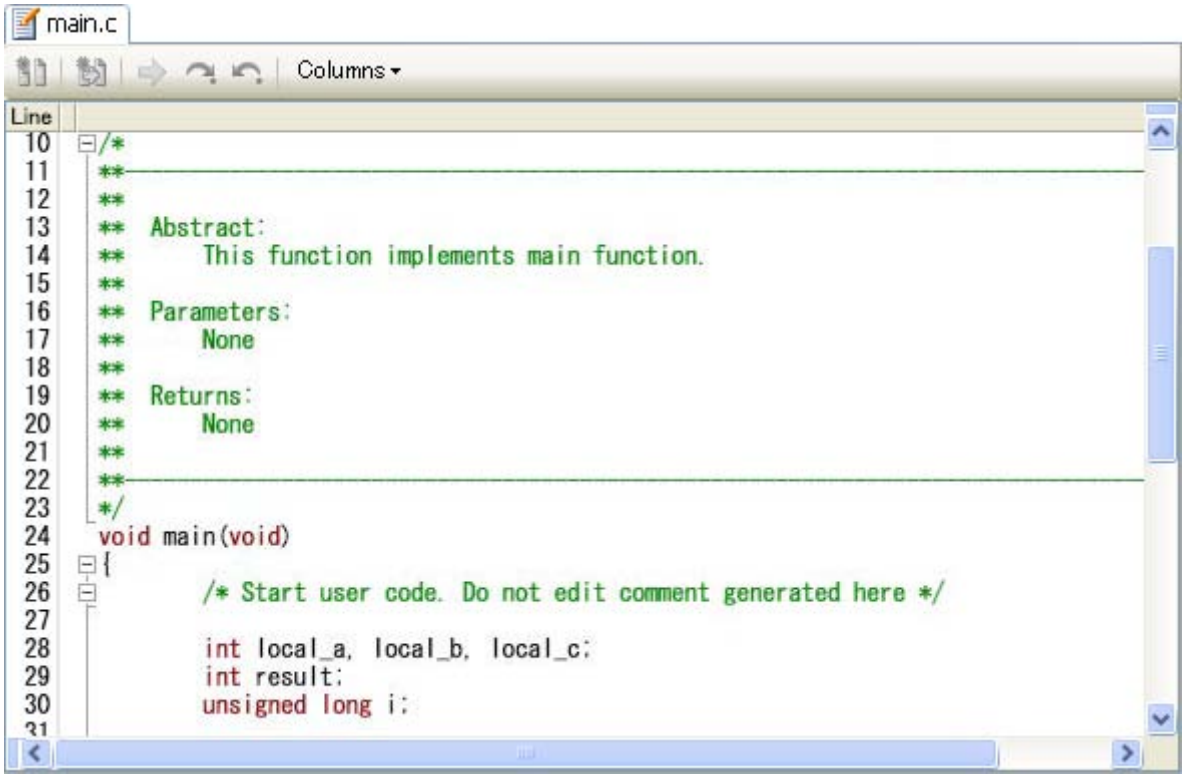
Memo	Add memos to the category of files. Add one item in one line. The specified memo is displayed as the subproperty.	
	Default	Memo[*number of items*]
	How to change	Edit by the Text Edit dialog box which appears when clicking the [...] button. For the subproperty, you can enter directly in the text box.
	Restriction	Up to 256 characters Up to 256 items can be specified.

Editor panel

This panel is used to display and edit text files and source files.

See "CubeSuite+ Integrated Development Environment User's Manual: RH850 Coding" for details about this panel.

Figure A-18. Editor Panel

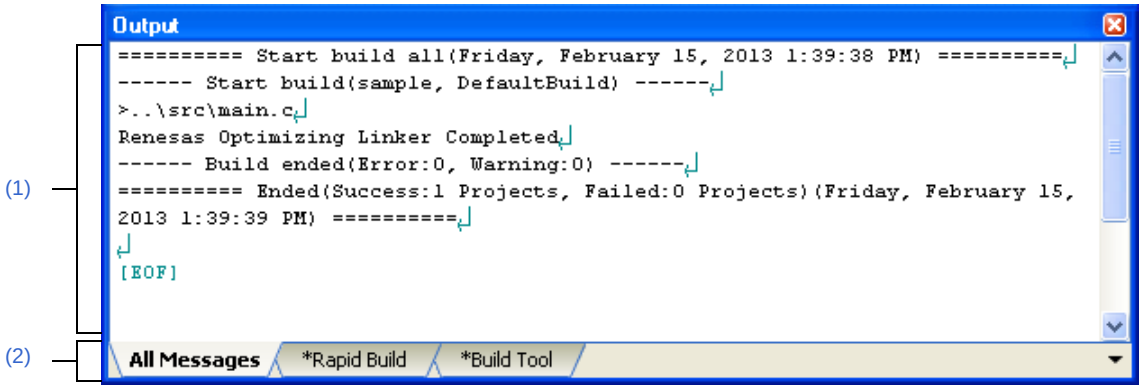


Output panel

This panel is used to display the message that is output from the build tool.
Messages are shown individually on the tab categorized by the output tool.

Remark This panel can be zoomed in and out by 100% in the tool bar, or by moving the mouse wheel forward or backward while holding down the [Ctrl] key.

Figure A-19. Output Panel



The following items are explained here.

- [How to open]
- [Description of each area]
- [[File] menu (only available for the Output panel)]
- [[Edit] menu (only available for the Output panel)]
- [Context menu]

[How to open]

- From the [View] menu, select [Output].

[Description of each area]

(1) Message area

This area displays messages output from each tool.
In build result display, a new message is displayed deleting the previous message every time build is done (but not the [All Messages] tab).

Remark Up to 500000 lines of messages can be displayed.
If 500001 lines or more of messages are output, then the excess lines are deleted, oldest first.

The message colors differ as follows depends on the type of the output message (the character color/background color is set in the [General - Font and Color] category in the Option dialog box).

Message Type	Example (Default)			Description
Normal message	AaBbCc	Character color	Black	Information on something.
		Background color	White	

Message Type	Example (Default)			Description
Warning message	AaBbCc	Character color	Blue	Warning for the operation.
		Background color	Normal color	
Error message	AaBbCc	Character color	Red	Fatal error or operation disabled because of an error in operation.
		Background color	Light gray	

This area has the following functions.

(a) Tag jump

When the output message is double-clicked, or the [Enter] key is pressed with the caret over the message, the Editor panel appears and the destination line number of the file is displayed.
You can jump to the line of the source file that generated the error from the error message output when building.

(b) Display help

If you select the [Help for Message] from the context menu or press the [F1] key when the cursor is on a line displaying the warning or error message, the help related to that line's message is displayed.

(c) Save log

The contents displayed on the currently selected tab can be saved in a text file (*.txt) by selecting [Save Output - *tab name* As...] from the [File] menu to open the Save As dialog box (messages on the tab that is not selected will not be saved).

(2) Tab selection area

Select tabs that messages are output from.
Tabs that are displayed are as follows.

Tab Name	Description
All Messages	Displays all the messages by order of output (except while executing a rapid build).
Rapid Build	Displays the message output from the build tool by running a rapid build.
Build Tool	Displays the message output from the build tool by running a build, rebuild, or batch build.

Caution Even if a new message is output on a deselected tab, tab selection will not automatically switch. In this case, a mark will be added in front of the tab name, indicating that a new message has been output.

[[File] menu (only available for the Output panel)]

The following items are exclusive for the [File] menu in the Output panel (other items are common to all the panels).

Save Output - *tab name*	Saves the contents on the currently selecting tab in the previously saved text file (*.txt) (see "(c) Save log"). When this item is selected for the first time after launching the program, the operation is equivalent to when selecting [Save Output - *tab name* As...].
Save Output - *tab name* As...	Opens the Save As dialog box to save the contents on the currently selecting tab in the designated text file (*.txt) (see "(c) Save log").

[[Edit] menu (only available for the Output panel)]

The following items are exclusive for the [Edit] menu in the Output panel (other items are all invalid).

Copy	Copies the selected characters to the clipboard.
Select All	Selects all the messages displayed on this panel.
Find...	Opens the Find and Replace dialog box with the [Quick Find] tab target.
Replace...	Opens the Find and Replace dialog box with the [Replace in Files] tab target.

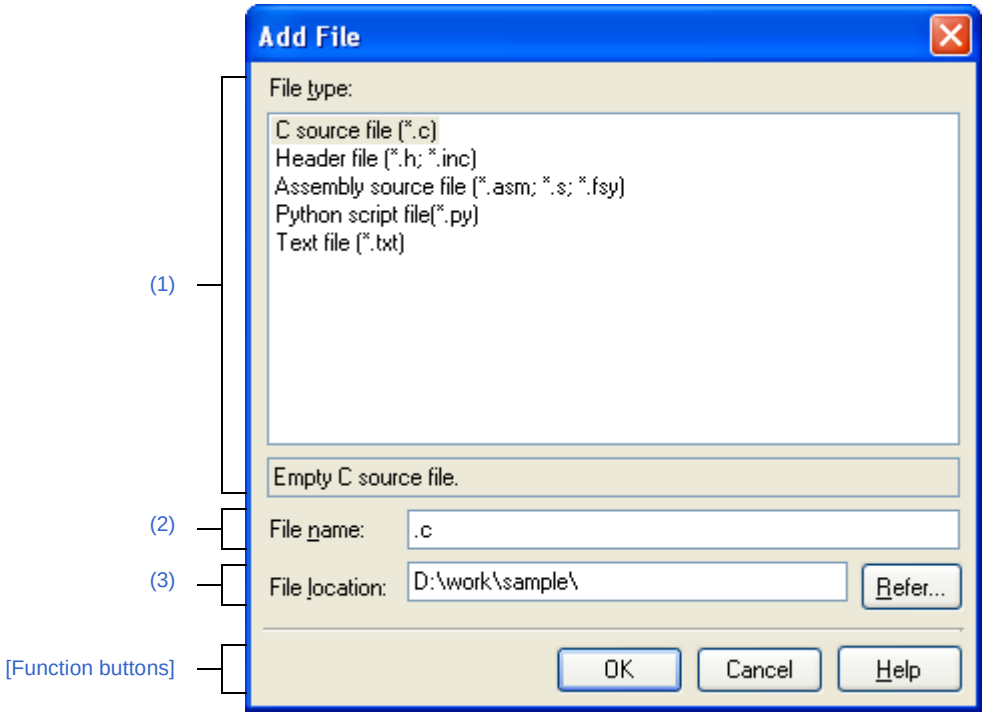
[Context menu]

Copy	Copies the selected characters to the clipboard.
Select All	Selects all the messages displayed on this panel.
Clear	Deletes all the messages displayed on this panel.
Tag jump	If there is information of the file name, line, and column on the caret line, jumps to that location.
Help for Message	Displays the help related to the message on the current caret position. This only applies to warning messages and error messages.

Add File dialog box

This dialog box is used to create a new file and add it to the project.

Figure A-20. Add File Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- From the [File] menu, select [Add] >> [Add New File...].
- On the Project Tree panel, select either one of the Project node, Subproject node, File node, or Category node, and then select [Add] >> [Add New File...] from the context menu.

[Description of each area]

(1) [File type] area

Select the type of the file to be created.

When the file type is selected, the description is displayed at the lower box.

The file types to be displayed are as follows.

- C source file (*.c)
- Header file (*.h; *.inc)
- Assembly source file (*.asm; *.s; *.fsy)
- Python script file (*.py)
- Text file (*.txt)

(2) [File name] area

Directly enter the name of the file to be created.
".txt" is displayed by default.

Remark If any extension is not designated, the one selected in the [File type] area will be added.
Also, if the extension different from the one selected in the [File type] area is designated, the one selected in the [File type] area will be added (for example, if you designate "aaa.txt" as the file name and select "C source file (*.c)" as the file type, the file is named as "aaa.txt.c").

(3) [File location] area

Designate the location to be created the file by directly entering its path or selecting from the [Refer...] button.
The path of the project folder is displayed by default.
Note, however, that the folder path set in the category is displayed when this dialog box is opened from the context menu of the category node (only when the shortcut to the folder has been set and the folder exists).

(a) Button

Refer...	Opens the Browse For Folder dialog box. If a folder is selected, the path will be added in the text box.

Remarks 1. If the text box is blank, it is assumed that the project folder is designated.
2. If the relative path is designated, the reference point of the path is the project folder.

Remark Up to 259 characters (path and file name combined) can be specified in the [File name] area and [File location] area. When the input violates any restriction, the following messages will be shown on the [File name] area in the tooltip.

Message	Description
The file name including the path is too long. Make it within 259 characters.	The file name with the path is more than 259 characters.
The specified path contains a folder that does not exist.	The path contains a folder that does not exist.
The file name or path name is invalid. The following characters cannot be used: \, /, :, *, ?, ", <, >, |	The file name with the invalid path is designated. The following characters cannot be used for the file name and folder name: \, /, :, *, ?, ", <, >, |

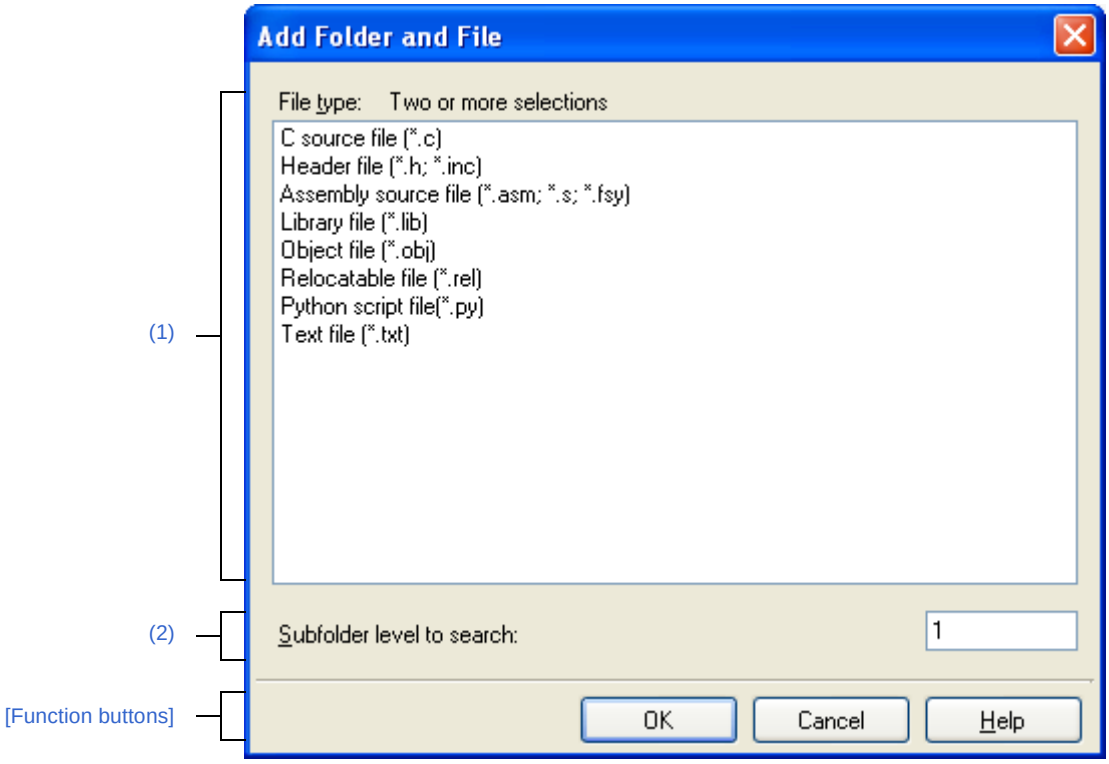
[Function buttons]

Button	Function
OK	Creates the file with the entered file name, adds it to the project, and opens with the Editor panel. And then closes this dialog box.
Cancel	Does not create a file and closes this dialog box.
Help	Displays the help of this dialog box.

Add Folder and File dialog box

This dialog box is used to add existing files and folder hierarchies to the project.
The folder is added as a category.

Figure A-21. Add Folder and File Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Drag the folder from Explorer or the like, and drop it on the Project Tree panel.

[Description of each area]

(1) [File type] area

Select the type of the file to be added to the project.
You can select multiple file types by left clicking while holding down the [Ctrl] or [Shift] key.
If nothing is selected, it is assumed that all types are selected.
The file types to be displayed are as follows.

- C source file (*.c)
- Header file (*.h; *.inc)
- Assembly source file (*.asm; *.s; *.fsy)
- Library file (*.lib)
- Object file (*.obj)

- Relocatable file (*.rel)
- Python script file (*.py)
- Text file (*.txt)

(2) [Subfolder level to search] area

Directly enter the number of levels of the subfolder to be added to the project.
"1" is displayed by default.

Remark Up to 10 (decimal number) can be specified.
When the input violates any restriction, the following messages will be shown in the tooltip.

Message	Description
Fewer than 0 or more than 10 values cannot be specified.	Fewer than 0 or more than 10 subfolder levels have been specified.
Specify in decimal.	A number in other than decimal or a string has been specified.

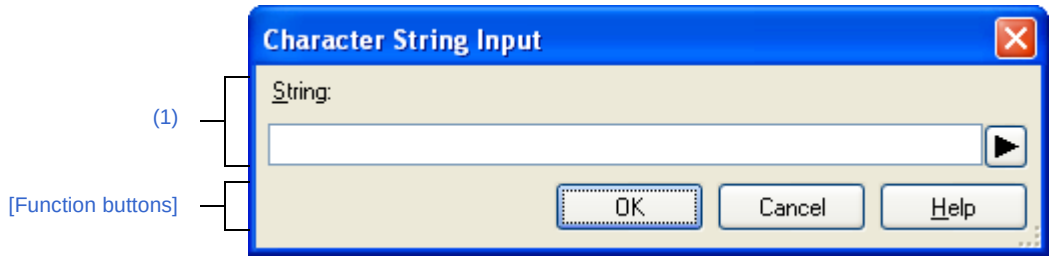
[Function buttons]

Button	Function
OK	Adds the folder that was dragged and dropped and the files in that folder to the project. And then closes this dialog box.
Cancel	Does not add the folder and files, and then closes this dialog box.
Help	Displays the help of this dialog box.

Character String Input dialog box

This dialog box is used to input and edit characters in one line.

Figure A-22. Character String Input Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Project Tree panel, select a file and then select [Change Extension...] from the context menu to open a message dialog box.
  And then click the [Yes] button in the dialog box.
- On the Property panel, select the following properties, and then click the [...] button.
  - From the [Common Options] tab, [Undisplayed warning message] in the [Warning Message] category, [Format of build option list], [Other additional options] in the [Others] category
  - From the [Compile Options] tab, [Other additional options] in the [Others] category
  - From the [Assemble Options] tab, [Other additional options] in the [Others] category
  - From the [Link Options] tab, [Execution start address] in the [Output Code] category, [Suppress number of information message], [Number of warning message], [Number of information message], [Number of information and warning message] in the [Message] category, and [Other additional options] in the [Others] category
  - From the [Hex Output Options] tab, [Other additional options] in the [Others] category
  - From the [Create Library Options] tab, [Suppress number of information message], [Number of warning message], [Number of information message], [Number of information and warning message] in the [Message] category, and [Other additional options] in the [Others] category
  - From the [Individual Compile Options] tab, [Undisplayed warning message] in the [Warning Message] category, and [Other additional options] property in the [Others] category
  - From the [Individual Assemble Options] tab, [Undisplayed warning message] in the [Warning Message] category, and [Other additional options] property in the [Others] category

[Description of each area]

(1) [String] area

Input characters in one line.
By default, the current value of the area that this dialog box is called from is reflected to this area.
You cannot start a new line.

Remark Up to 32767 characters can be entered.
When the input violates any restriction, the following messages will be shown in the tooltip.

Message	Description
More than *maximum number of restriction in the property that called this dialog box* characters cannot be specified.	The numbers of input characters exceeds the maximum number of restriction in the property that called this dialog box.

(a) Button

	The placeholders which can be specified for the area that this dialog box is called from is displayed in a popup (ascending order). If a placeholder is selected, the string will be surrounded with percentage signs ("%"), and displayed in [String].

Caution This button is displayed only when the caller of this dialog box supports placeholders.

Remark The placeholders which can be specified differ depending on the area that this dialog box is called from.
For the specific placeholder, see the description of the area that this dialog box is called from.

[Function buttons]

Button	Function
OK	Reflects the entered characters to the area that this dialog box is called from and closes this dialog box.
Cancel	Does not reflect the entered characters to the area that this dialog box is called from and closes this dialog box.
Help	Displays the help of this dialog box.

Text Edit dialog box

This dialog box is used to input and edit texts in multiple lines.

Figure A-23. Text Edit Dialog Box (When Caller Supports Placeholders)

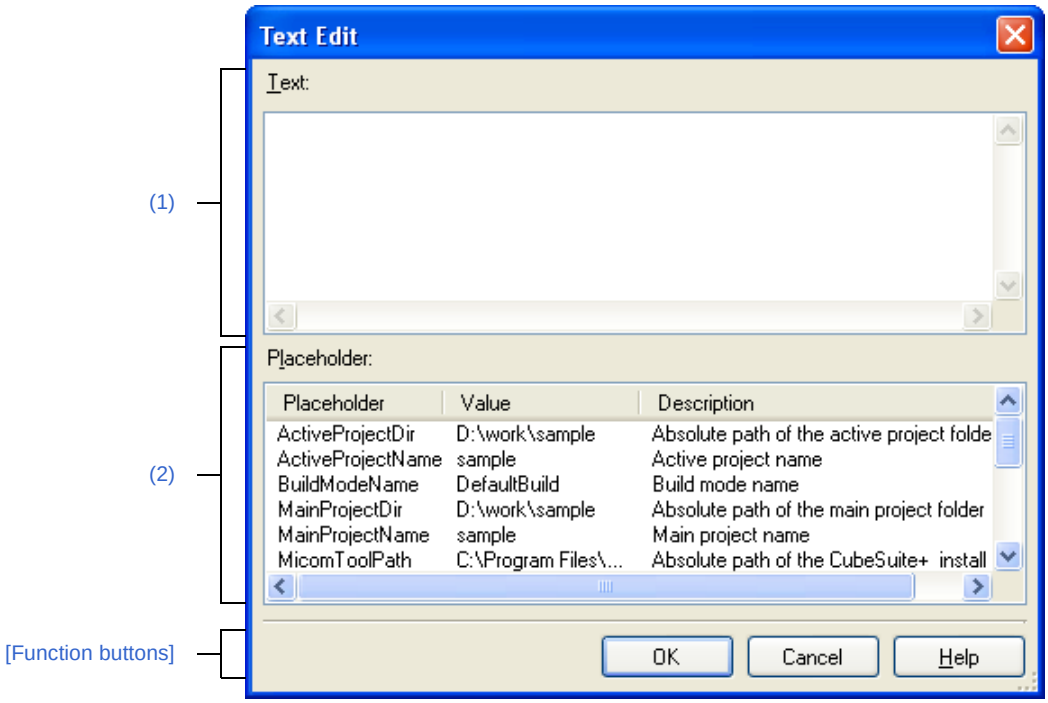
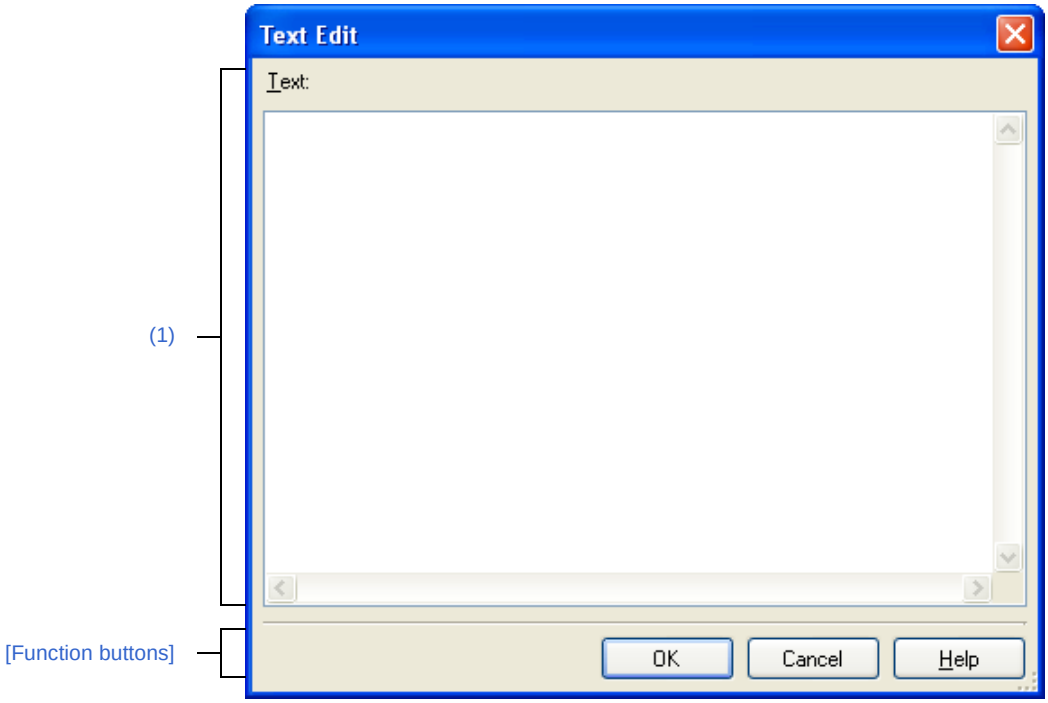


Figure A-24. Text Edit Dialog Box (When Caller Does Not Support Placeholders)



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Property panel, select the following properties, and then click the [...] button.
  - From the [Common Options] tab, [Macro definition] in the [Frequently Used Options(for Compile)] category, [Macro definition] in the [Frequently Used Options(for Assemble)] category, [Division output file] in the [Frequently Used Options(for Link)] category, [Division output file] in the [Frequently Used Options(for Hex Output)] category, [Memo] in the [Notes] category, and [Commands executed before build processing], [Commands executed after build processing] in the [Others] category
  - From the [Compile Options] tab, [Include files at head of compiling units], [Macro definition], [Macro undefinition] in the [Preprocess] category, and [Commands executed before compile processing], [Commands executed after compile processing] in the [Others] category
  - From the [Assemble Options] tab, [Macro definition], [Macro undefinition] in the [Preprocess] category, and [Commands executed before assemble processing], [Commands executed after assemble processing] in the [Others] category
  - From the [Link Options] tab, [Object file], [Binary file], [Symbol definition] in the [Input File] category, [Division output file] in the [Output File] category, [Section that includes startup function], [Section that outputs external defined symbols to the file], [Section alignment], [ROM to RAM mapped section] in the [Section] category, [Address range of memory type] in the [Verify] category, and [Commands executed before link processing], [Commands executed after link processing] in the [Others] category
  - From the [Hex Output Options] tab, [Division output file] in the [Output File] category
  - From the [Create Library Options] tab, [Object file], [Binary file] in the [Input File] category, and [Commands executed before create library processing], [Commands executed after create library processing] in the [Others] category
  - From the [Individual Compile Options] tab, [Include files at head of compiling units], [Macro definition], [Macro undefinition] in the [Preprocess] category, and [Commands executed before compile processing], [Commands executed after compile processing] in the [Others] category
  - From the [Individual Assemble Options] tab, [Macro definition], [Macro undefinition] in the [Preprocess] category, and [Commands executed before assemble processing], [Commands executed after assemble processing] in the [Others] category
  - From the [File Information] tab, [Memo] in the [Notes] category
  - From the [Category Information] tab, [Memo] in the [Notes] category

[Description of each area]

(1) [Text]

Edit texts in multiple lines.

By default, the current value of the area that this dialog box is called from is reflected in this area.

Remark Up to 65535 lines and 65535 characters can be entered.
When the input violates any restriction, the following message will be shown in the tooltip.

Message	Description
More than *maximum number of restriction in the property that called this dialog box* characters cannot be specified. The current number of characters is displayed between brackets at the beginning of the line in excess of the limit.	The numbers of input characters exceeds the maximum number of restriction in the property that called this dialog box.

(2) [Placeholder] area

The list of placeholders which can be specified for the area that this dialog box is called from is displayed (ascending order).

Double click a row to surround the placeholder with percentage signs ("%") and display it in [Text].

(a) [Placeholder]

This area displays the placeholder.

(b) [Value]

This area displays the string after replacement with the placeholder.

(c) [Description]

This area displays the description of the placeholder.

Caution This area is displayed only when the caller of this dialog box supports placeholders.

Remark The placeholders which can be specified differ depending on the area that this dialog box is called from. For the specific placeholder, see the description of the area that this dialog box is called from.

[Function buttons]

Button	Function
OK	Reflects the entered text to the text box that opened this dialog box and closes this dialog box.
Cancel	Does not reflect the entered text to the text box that opened this dialog box and closes this dialog box.
Help	Displays the help of this dialog box.

Path Edit dialog box

This dialog box is used to edit or add the path or file name including the path.

Figure A-25. Path Edit Dialog Box (When Editing Path)

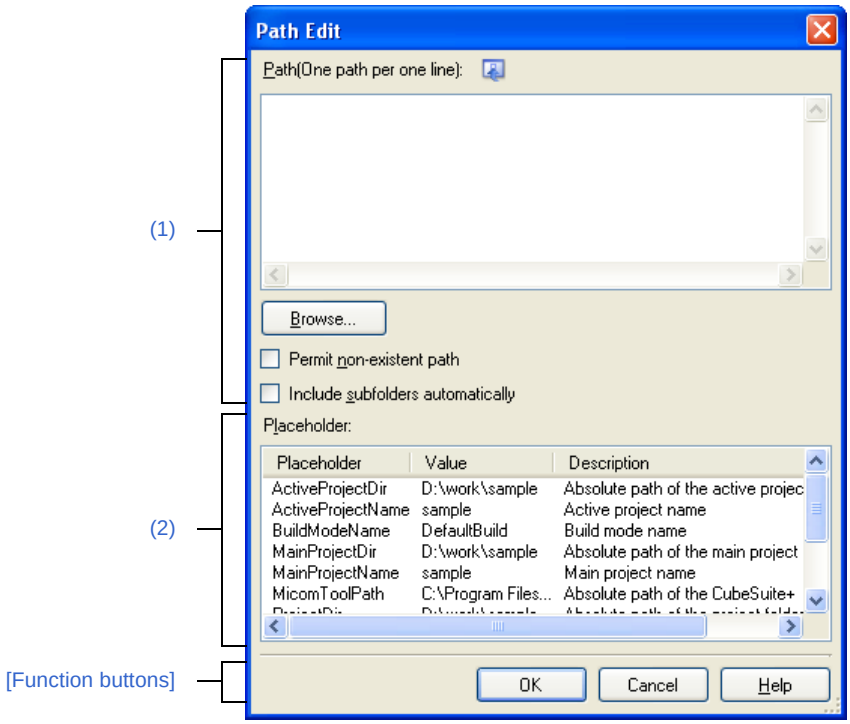
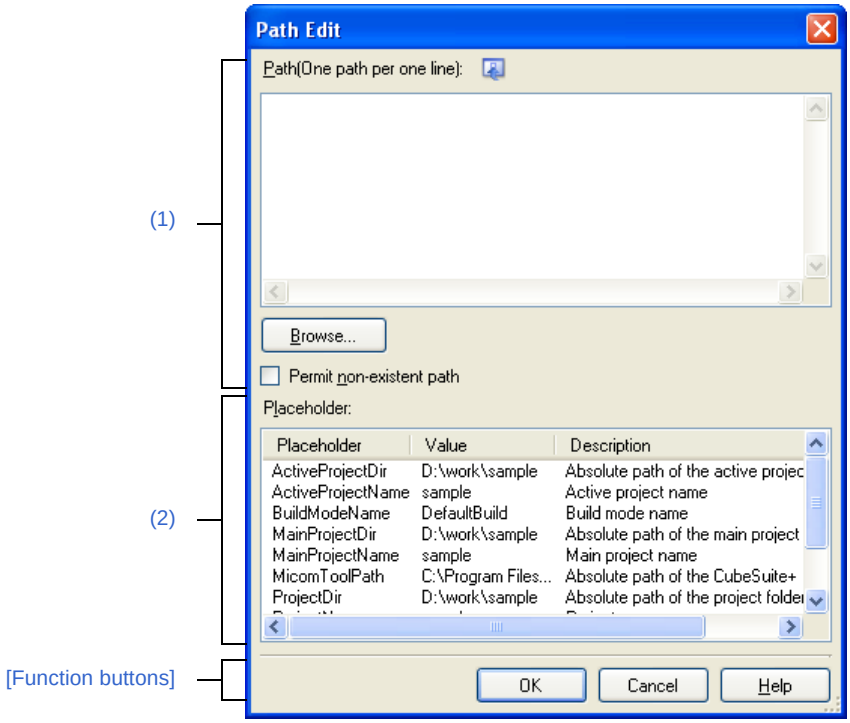


Figure A-26. Path Edit Dialog Box (When Editing File Name Including Path)



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Property panel, select the following properties, and then click the [...] button.
  - From the [Common Options] tab, [Additional include paths] in the [Frequently Used Options(for Compile)] category, [Additional include paths] in the [Frequently Used Options(for Assemble)] category, and [Using libraries] in the [Frequently Used Options(for Link)] category
  - From the [Compile Options] tab, [Additional include paths] in the [Preprocess] category, and [Rule check exclusion file] in the [MISRA-C:2004 Rule Check] category
  - From the [Assemble Options] tab, [Additional include paths] in the [Preprocess] category
  - From the [Link Options] tab, [Using libraries] in the [Library] category
  - From the [Create Library Options] tab, [Using libraries] in the [Library] category
  - From the [Individual Compile Options] tab, [Additional include paths] in the [Preprocess] category, and [Rule check exclusion file] in the [MISRA-C:2004 Rule Check] category
  - From the [Individual Assemble Options] tab, [Additional include paths] in the [Preprocess] category

[Description of each area]

(1) Path edit area

Edit or add the path or file name including the path.

(a) [Path(One path per one line)]

Edit or add the path or file name including the path by directly entering it.
The path or file name including the path can be designated in multiple lines. Designate the path or file name including the path at a line.
By default, the current contents of the text box that opened this dialog box are reflected in this area.

The path can be added by one of the following methods.

- Click the [Browse...] button, and then select the folder in the Browse For Folder dialog box.
- Drag and drop the folder using such as Explorer.

The file names including the path can be added by one of the following methods.

- Click the [Browse...] button, and then select the file in the Specify Far Jump File dialog box.
- Click the [Browse...] button, and then select the file in the Add Excluding File dialog box.
- Drag and drop the file using such as Explorer.

Caution **If an extremely long absolute path is specified as a relative path, an error could occur when clicking the [OK] button. In this case, designate the absolute path.**

Remark Up to 10000 lines can be entered. Up to the maximum characters that are limited by the Windows OS can be entered.
When the input violates any restriction, the following messages will be shown in the tooltip.

Message	Description
Specify a path.	The line contains space characters only.

Message	Description
The path is too long. Specify a path with a number of characters equal to or fewer than *maximum number of restriction in the property that called this dialog box*.	The file name including the path exceeds the maximum number of restriction in the property that called this dialog box.
The specified path contains a folder that does not exist.	The path contains a folder that does not exist.
The file name or path name is invalid. The following characters cannot be used: \, /, :, *, ?, ", <, >, |	The file name with the invalid path is designated. The following characters cannot be used for the file name and folder name: \, /, :, *, ?, ", <, >, |
More than *maximum number of paths or files specified by the caller* lines cannot be specified.	The number of paths or files which have been entered exceeds the maximum number of restriction in the property that called this dialog box.

(b) Button

Browse...	- When adding the path Opens the Browse For Folder dialog box. If a folder is selected, the path will be added in [Path(One path per one line)]. - When adding the file name including the path Opens the Specify Far Jump File dialog box. If a file is selected, the file will be added in [Path(One path per one line)].

(c) [Permit non-existent path]

When this check box is selected, the existence of the path specified in [Path(One path per one line)] or the validity of the character string specified in the path is not checked.

(d) [Include subfolders automatically]

Select this check box and then click the [Browse...] button to specify the path. The path will be added, including subfolders, to [Path(One path per one line)] (up to five levels deep).

Caution This item is displayed only when adding the path.

(2) [Placeholder] area

The list of placeholders which can be specified for the area that this dialog box is called from is displayed (ascending order).
Double click a row to surround the placeholder with percentage signs ("%") and display it in the path edit area.

(a) [Placeholder]

This area displays the placeholder.

(b) [Value]

This area displays the string after replacement with the placeholder.

(c) [Description]

This area displays the description of the placeholder.

Caution This area is displayed only when the caller of this dialog box supports placeholders.

Remark The placeholders which can be specified differ depending on the area that this dialog box is called from. For the specific placeholder, see the description of the area that this dialog box is called from.

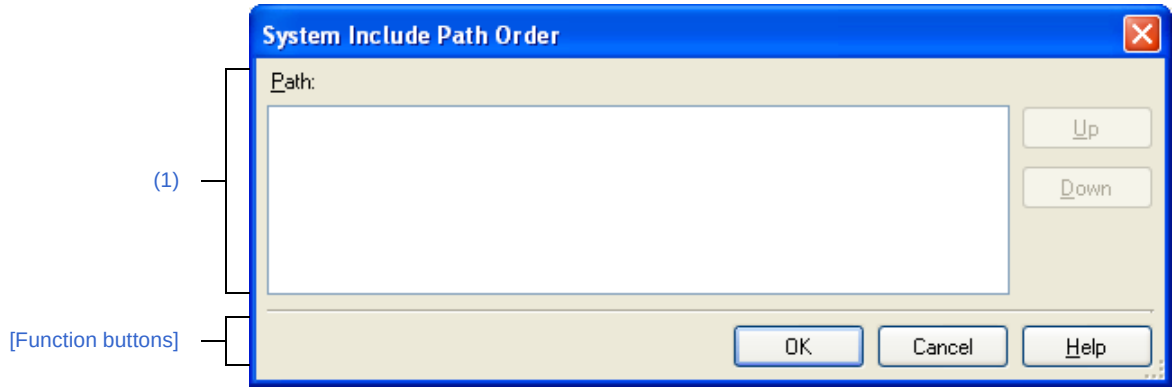
[Function buttons]

Button	Function
OK	Reflects the entered path to the text box that opened this dialog box and closes this dialog box.
Cancel	Does not reflect the entered path to the text box that opened this dialog box and closes this dialog box.
Help	Displays the help of this dialog box.

System Include Path Order dialog box

This dialog box is used to refer the system include paths specified for the compiler and set their specified sequence.

Figure A-27. System Include Path Order Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Property panel, select the following properties, and then click the [...] button.
  - From the [Common Options] tab, [System include paths] in the [Frequently Used Options(for Compile)] category, and [System include paths] in the [Frequently Used Options(for Assemble)] category
  - From the [Compile Options] tab, [System include paths] in the [Preprocess] category
  - From the [Assemble Options] tab, [System include paths] in the [Preprocess] category

[Description of each area]

(1) Path list display area

This area displays the list of the system include paths specified for the compiler.

(a) [Path]

This area displays the list of the system include paths in the specified sequence for the compiler.
The default order is the order that the files are registered to the project.
By changing the display order of the paths, you can set the specified order of the paths to the compiler.
To change the display order, use the [Up] and [Down] buttons, or drag and drop the path names.

Remarks 1. Move the mouse cursor over a file name to display a tooltip with the absolute path of that file.

2. Newly added system include paths are added next to the last path of the list.

3. When the path names are dragged and dropped, the multiple path names which are next to each other can be selected together.

(b) Button

Up	Moves the selected path to up.

Down	Moves the selected path to down.

Remark Note that above buttons are disabled when any path is not selected.

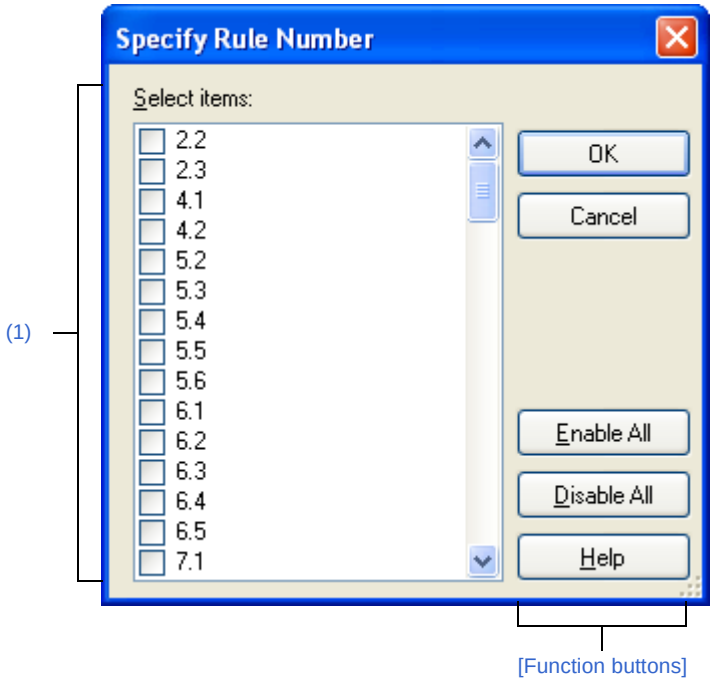
[Function buttons]

Button	Function
OK	Sets the specified order of the paths to the compiler as the display order in the Path list display area and closes this dialog box.
Cancel	Cancels the specified order of the paths and closes the dialog box.
Help	Displays the help of this dialog box.

Specify Rule Number dialog box

This dialog box is used to select the number of the MISRA-C:2004 rule and set it to the area that this dialog box is called from.

Figure A-28. Specify Rule Number Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Property panel, select the following properties, and then click the [...] button.
  - From the [Compile Options] tab, [Rule number], [Exclusion rule number], [Check rule number besides required rule], [Exclusion rule number from required rule] in the [MISRA-C:2004 Rule Check] category
  - From the [Individual Compile Options] tab, [Rule number], [Exclusion rule number], [Check rule number besides required rule] [Exclusion rule number from required rule] in the [MISRA-C:2004 Rule Check] category

[Description of each area]

(1) [Select items]

The list of the MISRA-C:2004 rule numbers which can be specified for the area that this dialog box is called from is displayed (ascending order).
Select the check boxes to set the rule number.

Remark In the area that this dialog box is called from, if a rule number is already set, the check box for that rule number will be selected by default.

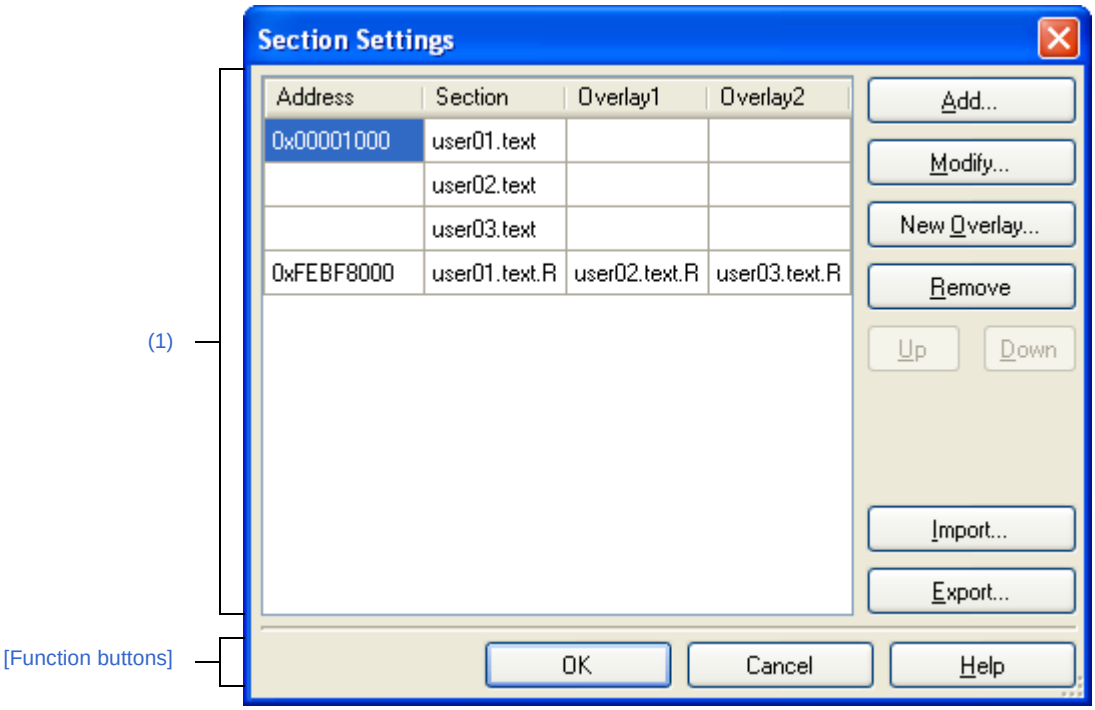
[Function buttons]

Button	Function
OK	Closes this dialog box and sets the selected rule number to the area that this dialog box is called from.
Cancel	Cancels the rule number selecting and closes the dialog box.
Enable All	Selects all the checkboxes in [Select items].
Disable All	Clears all the checkboxes in [Select items].
Help	Displays the help of this dialog box.

Section Settings dialog box

This dialog box is used to add, modify, or delete sections.

Figure A-29. Section Settings Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Property panel, select the following property, and then click the [...] button.
  - From the [Link Options] tab, [Section start address] in the [Section] category

[Description of each area]

(1) Address-section area

This area displays the list of currently configured section allocations.

(a) [Address]

This area displays the start addresses of the sections.

(b) [Section]

This area displays the names of the sections.

(c) [Overlay*n*]

This area displays the names of the sections to be overlaid (*n*: number starting with "1").

(d) Button

Add...	- When selecting an address in this area Opens the Section Address dialog box. Adds the address specified in the dialog box to this area so that the addresses are listed in the ascending order (the section column remains empty). - When selecting a section in this area Opens the Add Section dialog box. Adds the section specified in the dialog box to this area. When there is no empty column in the section group (an address and the sections allocated to the address) where the specified section is to be included, a new section row is added to the bottom of the section group. When there is an empty column, the section is added there.
Modify...	- When selecting an address in this area Opens the Section Address dialog box. Moves the section group according to the address specified in the dialog box so that the addresses are listed in the ascending order in this area. - When selecting a section in this area Opens the Modify Section dialog box. Replaces the section name selected in this area with the one specified in the dialog box. Note that this button is disabled when the selected sell is blank.
New Overlay...	Opens the Add Overlay dialog box. Adds the [Overlay*n*] column in this area and sets the section specified in the dialog box in the column that corresponds to the selected section group.
Remove	- When selecting an address in this area Opens the Unassigned Section dialog box. Deletes the section selected in the dialog box from this area. If no sections are left in the section group, the section group itself is deleted. - When selecting a section in this area Deletes the selected section from this area. If no sections are left in the section group, the section group itself is deleted. If no section names are left in the [Overlay*n*] column, the column itself is deleted. Note that this button is disabled when the selected sell is blank.
Up	Moves up the selected section. Note that this button is disabled when an address is selected.
Down	Moves down the selected section. Note that this button is disabled when an address is selected.
Import...	Opens the Select Import File dialog box. Acquires the section settings from the file specified in the dialog box and updates this area to reflect the acquired settings.
Export...	Opens the Select Export File dialog box. Outputs the contents of this area to the file specified in the dialog box.

[Function buttons]

Button	Function
OK	Reflects the specified section to the text box that opened this dialog box and closes this dialog box.
Cancel	Cancels the settings and closes this dialog box.
Help	Displays the help of this dialog box.

Add Section dialog box

Modify Section dialog box

Add Overlay dialog box

These dialog boxes are used to set a section name when adding, modifying, or overlaying a section, respectively.

Figure A-30. Add Section Dialog Box

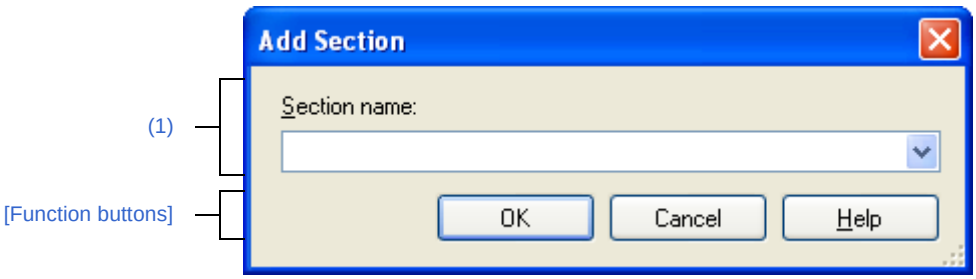


Figure A-31. Modify Section Dialog Box

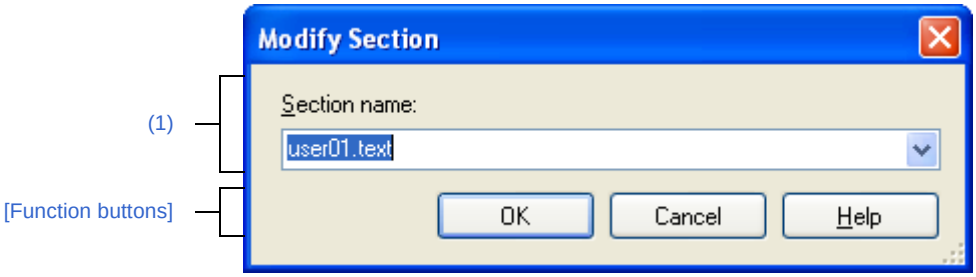
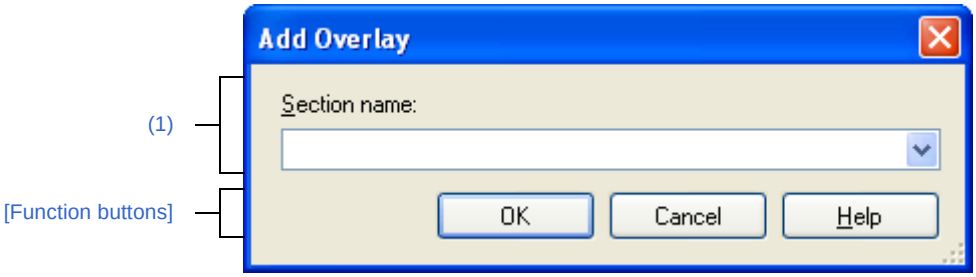


Figure A-32. Add Overlay Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Add Section dialog box
  - On the Section Settings dialog box, select a section in the address-section area, and then click the [Add...] button.
- Modify Section dialog box
  - On the Section Settings dialog box, select a section in the address-section area, and then click the [Modify...] button.
- Add Overlay dialog box
  - On the Section Settings dialog box, click the [New Overlay...] button.

[Description of each area]

(1) [Section name]

Specify the section name.

Directly enter the section name in the text box or select from the drop-down list.

The following characters can be used only: A-Z, a-z, 0-9, _, $, ., ~.

Wildcard characters (*) can also be used.

Note that numeric characters (0 to 9) cannot be used at the beginning of a section name.

The following reserved sections are set in the drop-down list.

.bss, .const, .data, .text

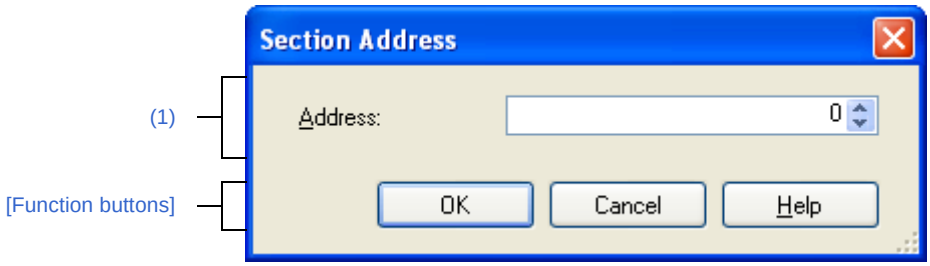
[Function buttons]

Button	Function
OK	- Add Section dialog box Closes this dialog box and adds the specified section to the address-section area in the Section Settings dialog box. When there is no empty column in the section group (an address and the sections allocated to the address) where the specified section is to be included, a new section row is added to the bottom of the section group.When there is an empty column, the section is added there. - Modify Section dialog box Closes this dialog box and replaces the section name selected in the address-section area in the Section Settings dialog box with the one specified. - Add Overlay dialog box Closes this dialog box and adds the [Overlay*n*] column (*n*: number starting with "1") to the address-section area in the Section Settings dialog box. Sets the specified section in the column that corresponds to the selected section group.
Cancel	Cancels the settings and closes this dialog box.
Help	Displays the help of this dialog box.

Section Address dialog box

This dialog box is used to set an address when adding or modifying a section.

Figure A-33. Section Address Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Section Settings dialog box, select an address in the address-section area, and then click the [Add...] or [Modify...] button.

[Description of each area]

(1) [Address]

Specify the start address of the section.
Directly enter the address in the text box or select from the button.
The range that can be specified for the value is 0 to FFFFFFFF (hexadecimal number) (default: 0).

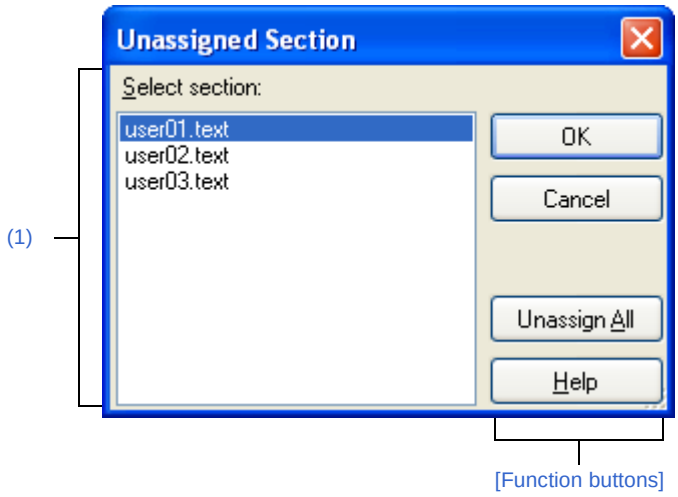
[Function buttons]

Button	Function
OK	- When opening from the [Add...] button in the Section Settings dialog box Closes this dialog box and adds the specified address to an appropriate location in the address-section area in the Section Settings dialog box (the section column remains empty). - When opening from the [Modify...] button in the Section Settings dialog box Closes this dialog box and moves the section group (an address and the sections allocated to the address) to an appropriate location in the address-section area in the Section Settings dialog box.
Cancel	Cancels the settings and closes this dialog box.
Help	Displays the help of this dialog box.

Unassigned Section dialog box

This dialog box is used to delete sections.

Figure A-34. Unassigned Section Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Section Settings dialog box, select an address in the address-section area, and then click the [Remove] button.

[Description of each area]

(1) [Select sections]

This area displays the name of all sections allocated to the address selected in the Section Settings dialog box.
Select sections to be deleted by clicking their names.
You can select multiple sections by left clicking while holding down the [Ctrl] or [Shift] key.

[Function buttons]

Button	Function
OK	Closes this dialog box and deletes the selected section from the address-section area in the Section Settings dialog box. Deletes the section group when the section group (an address and the sections allocated to the address) includes no section. If no sections are left in the [Overlay*n*] column in the address-section area, the column itself is deleted.
Cancel	Cancels the settings and closes this dialog box.

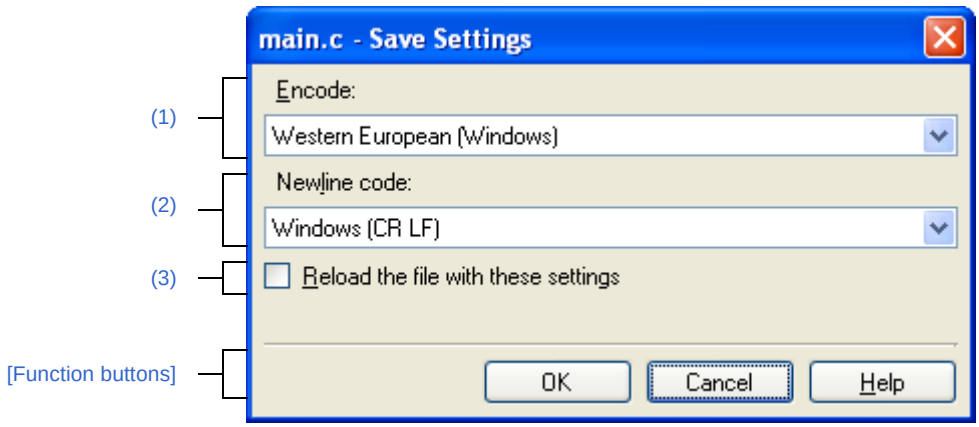
Button	Function
Unassign All	Closes this dialog box and deletes all the sections (the section group selected in the address-section area in the Section Settings dialog box).
Help	Displays the help of this dialog box.

Save Settings dialog box

This dialog box is used to set the encoding and newline code of the file that is being edited on the Editor panel.

Remark The target file name is displayed on the title bar.

Figure A-35. Save Settings Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Focus the Editor panel, and then select [*file name* Save Settings...] from the [File] menu.

[Description of each area]

(1) [Encode]

Select the encoding to be set from the drop-down list.

The items of the drop-down list are displayed according to the following sequence.

Note that the same encoding and encoding which are not supported by the current OS will not be displayed.

- *Current encoding of the file (default)*
- *Default encoding of the current OS*
- *Most recently used encodings (maximum 4)*
- *Popular encodings for current locale*

  (e.g. for United States locale it will be:

  - Western European (Windows)
  - Unicode (UTF-8)
- *All other encodings supported by the OS (in alphabetical order)*

(2) [Newline code]

Select the newline code to be set from the drop-down list.

You can select any of items below.

- Windows (CR LF)
- Macintosh (CR)

- Unix (LF)

An active newline entry is selected by default.

(3) [Reload the file with these settings]

☑	Reloads the file with the selected encoding and newline code when the [OK] button is clicked.
☐	Does not reload the file when the [OK] button is clicked (default).

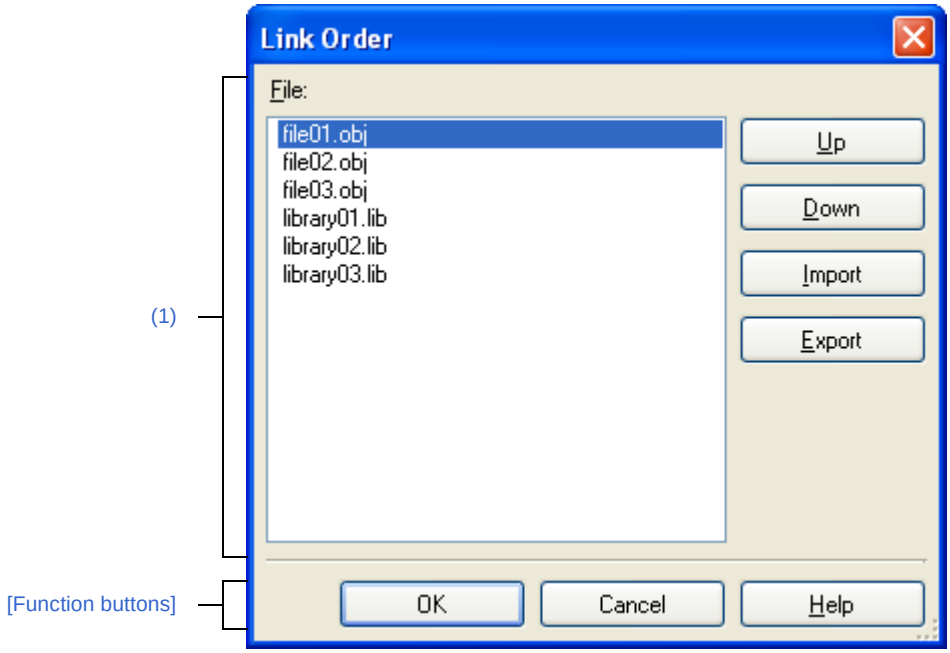
[Function buttons]

Button	Function
OK	Sets the selected encoding and newline code to the target file and closes this dialog box. If [Reload the file with these settings] is selected, sets the selected encoding and newline code to the target file and reloads the file. And then closes this dialog box.
Cancel	Cancels the settings and closes this dialog box.
Help	Displays the help of this dialog box.

Link Order dialog box

This dialog box is used to refer object module files and library files to be input to the linker and configure these link orders.

Figure A-36. Link Order Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Project Tree panel, select the Build tool node, and then select [Set Link Order...] from the context menu.

[Description of each area]

(1) [File] area

The names of the following files are listed in the order that the files are input to the linker.

- Object module files which are generated from the source files added to the selected main project or subproject
- Object module files which are added directly to the project tree of the selected main project or subproject
- Library files which are added directly to the project tree of the selected main project or subproject

The default order is the order that the files are added to the project.
By changing the display order of the files, you can set the input order of the files to the linker.
To change the display order, use the [Up] and [Down] buttons, or drag and drop the file names.

Remarks 1. When the mouse cursor is hovered over a file name, the path of the file appears in a popup. If the file is on the same drive as the project file, then it appears as the relative path; if it is on the different drive, then it appears as the absolute path.

2. Object module files which are generated from newly added source files and newly added object module files are added after the last object module file in the list. Newly added library files are added to the end of the list.
3. When the file is dragged and dropped, the multiple files that are next to each other can be selected together.

(a) Button

Up	Moves the selected file to up. If any file is not selected, this button will be disabled.
Down	Moves the selected file to down. If any file is not selected, this button will be disabled.
Import	Opens the Select Import File dialog box. The description order of the file names are acquired from the selected link order specification file, and then they are reflected in [File]. If nothing is displayed in [File], this button will be disabled.
Export	Opens the Select Export File dialog box. Outputs the list of the file names diaplayed in [File] to the specifiled link order specification file. If nothing is displayed in [File], this button will be disabled.

Remark See "2.11.2 Set the link order of files" for the method of using the link order specification file.

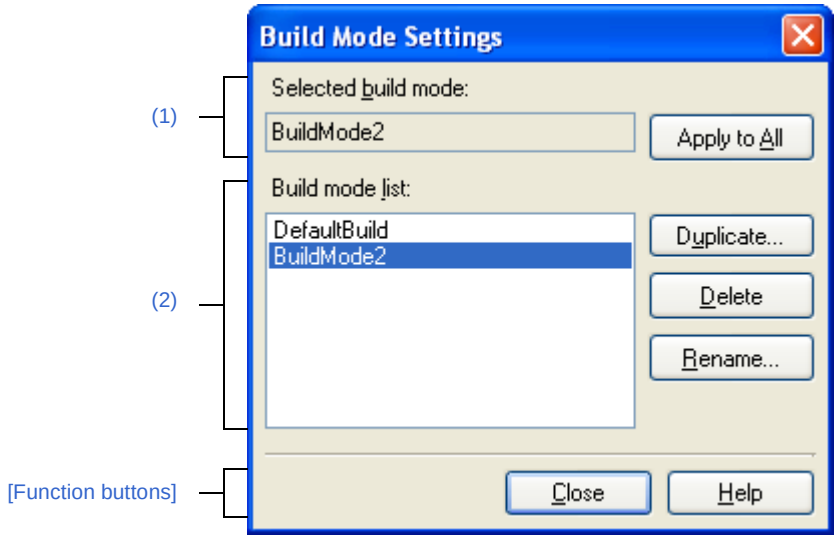
[Function buttons]

Button	Function
OK	Sets the input order of the files to the linker as the display order in [File] and closes this dialog box.
Cancel	Cancels the link order settings and closes the dialog box.
Help	Displays the help of this dialog box.

Build Mode Settings dialog box

This dialog box is used to add and delete build modes and configure the current build mode in batch.

Figure A-37. Build Mode Settings Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- From the [Build] menu, select [Build Mode Settings...].

[Description of each area]

(1) [Selected build mode] area

This area displays the build mode selected in the [Build mode list] area.

(a) Button

Apply to All	Set the build mode of the main project and all subprojects of the currently opened project to the currently displayed build mode.

(2) [Build mode list] area

This area displays all build modes that exist in the currently opening project (main project and subproject) in a list.
The current build modes of all projects are same, the build mode is selected by default.If they are not same, "DefaultBuild" will be selected.
The build mode that exists only in part of the main project and subproject is shown with the mark "*".
Note that the "DefaultBuild" is the default build mode and is always displayed at the top.

(a) Button

Duplicate...	Duplicates the selected build mode. The Character String Input dialog box opens. And the build mode is duplicated with the name entered in the dialog box and added to the main project and all the subprojects in the currently opening project. When the build mode with "*" mark does not exist in the main project or subproject and duplicate the build mode, "DefaultBuild" will be duplicated. Up to 20 build modes can be added.
Delete	Deletes the selected build mode. Note that "DefaultBuild" cannot be deleted.
Rename...	Renames the selected build mode. Rename the build mode with entered name in the opening the Character String Input dialog box.

Caution When duplicating or renaming the build mode, the existing build mode name cannot be used.

Remarks 1. Up to 127 characters can be specified as a build mode name. When the input violates any restriction, the following messages are shown in the tooltip.

Message	Description
A build mode with the same name already exists.	The entered build mode name already exists.
More than 127 characters cannot be specified.	Build mode name is too long (more than 128 characters).
The build mode name is invalid. The following characters cannot be used: \, /, :, *, ?, ", <, >, |	Invalid build mode name is entered. The characters (\, /, :, *, ?, ", <, >, |) cannot be used because the build mode name is used for the folder name.

2. Up to 20 build modes can be added. When the input violates any restriction, the following messages are shown in the tooltip.

Message	Description
The maximum number of build modes that can be set per project/subproject is 20.	The number of build modes exceed 20.

[Function buttons]

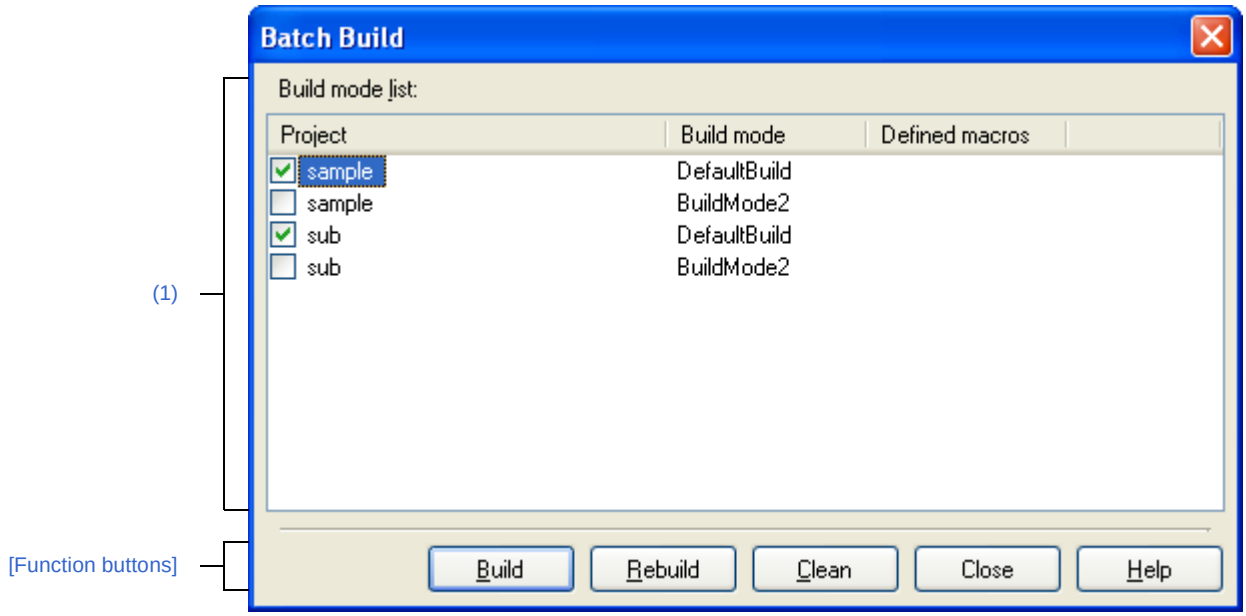
Button	Function
Close	Closes this dialog box.
Help	Displays the help of this dialog box.

Batch Build dialog box

This dialog box is used to run builds, rebuilds and cleans in batch with the build modes that the project (main project and subproject) has.

Remark The batch build order follows the project build order, the order of the subprojects, main project.
When multiple build modes are selected for a single main project or subproject, after running builds of the subproject with all the selected build modes, the build of the next subproject or main project is run.

Figure A-38. Batch Build Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- From the [Build] menu, select [Batch Build...].

[Description of each area]

(1) [Build mode list] area

This area displays the list of the combinations of the names of the main project and subprojects in the currently opened project, their build modes, and their macro definitions.

(a) [Project]

This area displays the main project and subprojects in the currently opened project.
Select the check boxes for the combinations of the main project and subprojects and their build modes which you wish to run a build.
When this dialog box is opened for the first time after the project is created, all the check boxes are unchecked. From the second time, the previous setting is retained.

(b) [Build mode]

This area displays the build modes which the main project and subprojects have.

(c) [Defined macros]

For the combination of the main project and subprojects and their build modes, the defined macros which have been set in the [Compile Options] tab and the [Assemble Options] tab on the Property panel are separated with "|" and displayed.

The defined macro in the compile option comes before the one in assemble option. They are separated with ", " and displayed.

Remarks 1. You can select multiple lines by one of the following procedures.

- Left-click the mouse with pressing the [Ctrl] key.
- Left-click the mouse with pressing the [Shift] key.
- Drag the mouse to select a range of rows.

Pressing the space key while a row is selected switches the selected/non-selected state of the checkbox.

2. Clicking on the header of each column sorts the entries (in ascending order by default). Clicking on the same header again sorts the entries in the reverse order.

[Function buttons]

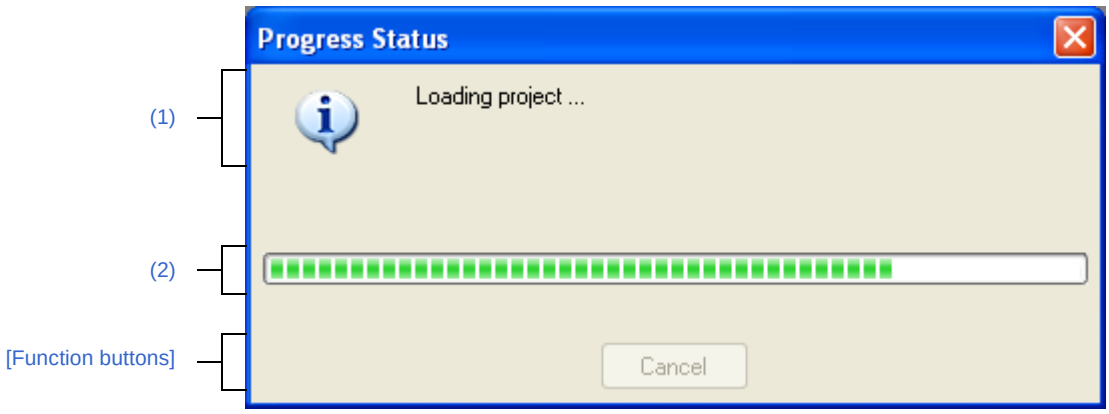
Button	Function
Build	Closes this dialog box and runs builds of the selected projects in the respective build modes. The execution result of the builds is displayed on the Output panel. After the builds are complete, the build mode configuration restores to the one before this dialog box was opened. If any project is not selected, this button will be disabled.
Rebuild	Closes this dialog box and runs rebuilds of the selected projects in the respective build modes. The execution result of the rebuilds is displayed on the Output panel. After the rebuilds are complete, the build mode configuration restores to the one before this dialog box was opened. If any project is not selected, this button will be disabled.
Clean	Closes this dialog box and deletes the files which are built in the respective build modes set for the selected projects. The execution result of the cleans is displayed on the Output panel. After the cleans are complete, the build mode configuration restores to the one before this dialog box was opened. If any project is not selected, this button will be disabled.
Close	Closes this dialog box.
Help	Displays the help of this dialog box.

Progress Status dialog box

This dialog box is used to show how the process has been progressed when the time consuming process is taken place.

This dialog box automatically closes when the process in progress is done.

Figure A-39. Progress Status Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- This dialog box automatically opens when a message is output while the time consuming process is in progress.

[Description of each area]

(1) Message display area

This area displays messages output while process is in progress (edit not allowed).

(2) Progress bar

The progress bar shows the current progress of the process in progress with the bar length.

When the process is 100% done (the bar gets to the right end), this dialog box automatically closed.

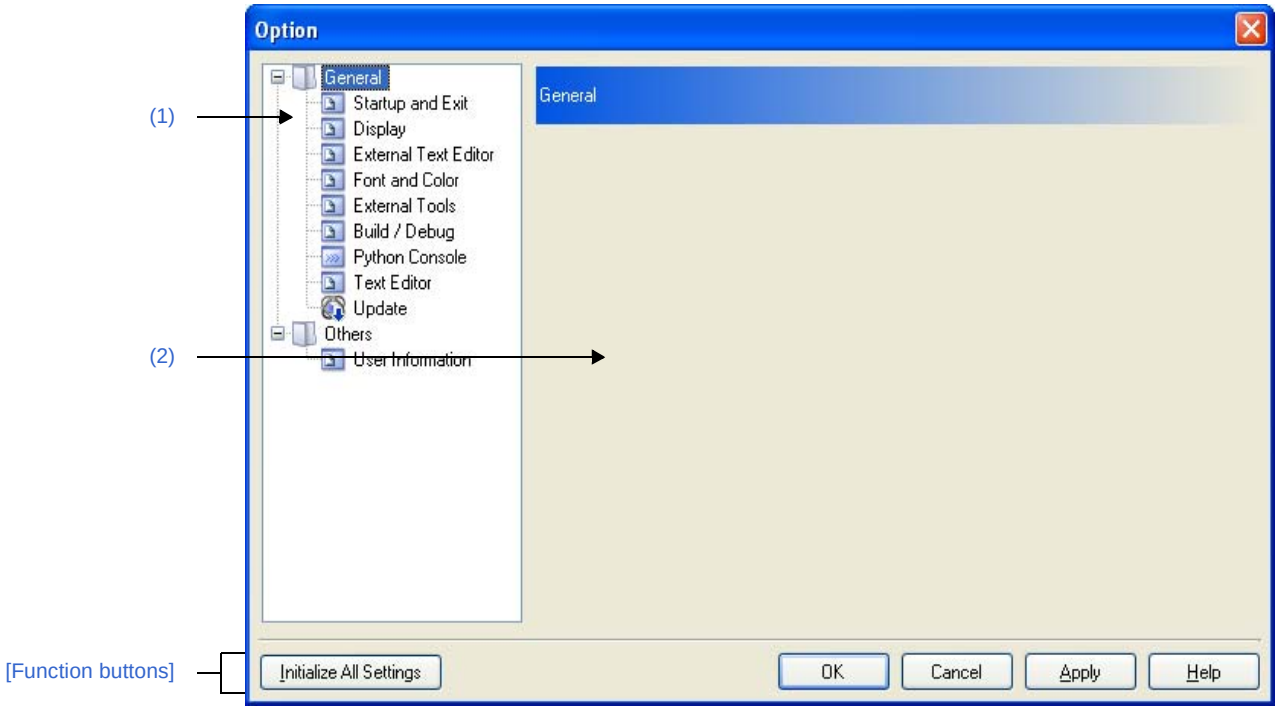
[Function buttons]

Button	Function
Cancel	Cancels the process in progress and closes this dialog box. Note that this button will be disabled if the process termination is impossible.

Option dialog box

This dialog box is used to configure the CubeSuite+ environment.
All settings made via this dialog box are saved as preferences for the current user.

Figure A-40. Option Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- From the [Tool] menu, select [Options...].

[Description of each area]

(1) Category selection area

Select the items to configure from the following categories.

Category	Description
[General - Startup and Exit] category	Configure startup and shutdown.
[General - Display] category	Configure messages from the application.
[General - External Text Editor] category	Configure the external text editor.
[General - Font and Color] category	Configure the fonts and colors shown on each panel.
[General - External Tools] category	Configure the startup of external tools.
[General - Build/Debug] category	Configure building and debugging.

Category	Description
[General - Python Console] category	Configure the Python console.
[General - Text Editor] category	Configure the text editor.
[General - Update] category	Configure update.
[Other - User Information] category	Configure user information.

Remark See "CubeSuite+ Integrated Development Environment User's Manual: Start" for details about categories other than [General - Build/Debug].

(2) Settings

This area is used to configure the various options for the selected category.
For details about configuration for a particular category, see the section for the category in question.

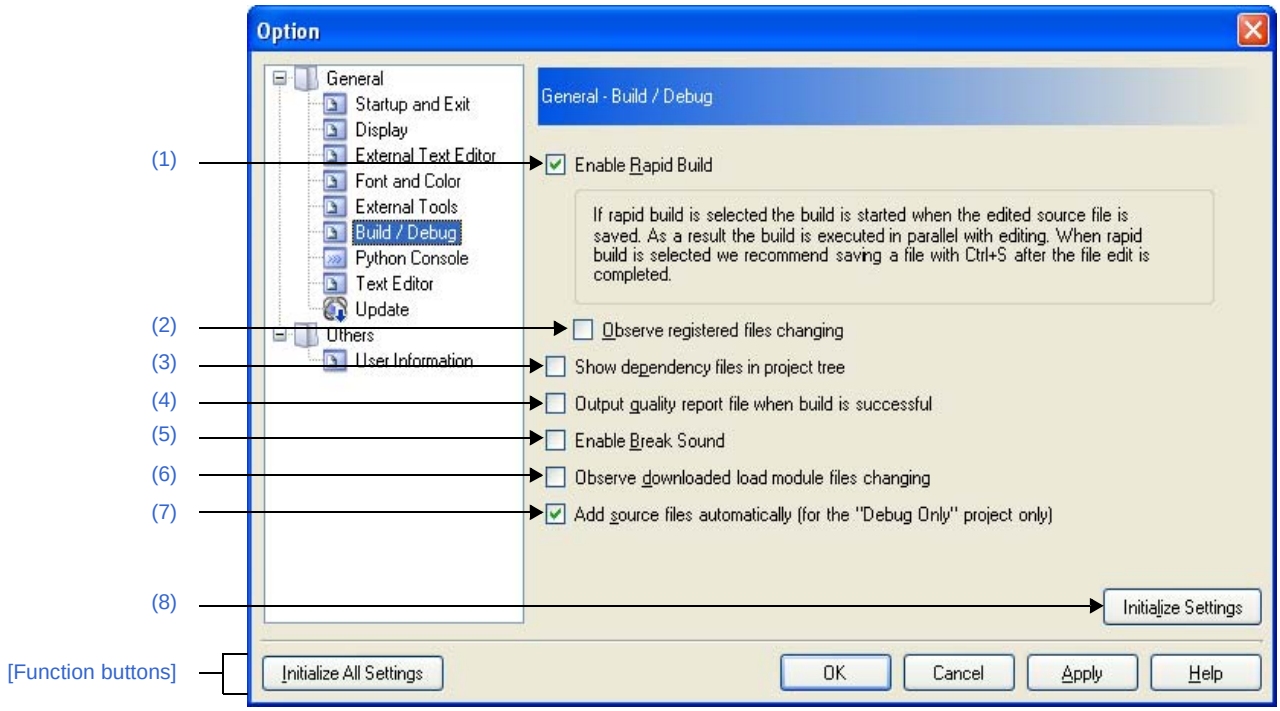
[Function buttons]

Button	Function
Initialize All Settings	Restores all settings on this dialog box to their default values. Note, however, that newly added items in the [General - External Tools] category will not be removed.
OK	Applies all setting and closes this dialog box.
Cancel	Ignores the setting and closes this dialog box.
Apply	Applies all setting (does not close this dialog box).
Help	Displays the help of this dialog box.

[General - Build/Debug] category

Use this category to configure general setting relating to building and debugging.

Figure A-41. Option Dialog Box ([General - Build/Debug] Category)



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- From the [Tool] menu, select [Options...].

[Description of each area]

(1) [Enable Rapid Build]

☑	Enables the rapid build Note feature (default).
☐	Does not use the rapid build feature.

Note This feature automatically begins a build when the source file being edited is saved.
Enabling this feature makes it possible to perform builds while editing source files.
If this feature is used, we recommend saving frequently after editing source files.

(2) [Observe registered files changing]

This item is only enabled if the [Enable Rapid Build] check box is selected.

☑	Starts a rapid build when a source file registered in the project is edited or saved by an external text editor or the like.
☐	Does not start a rapid build when a source file registered in the project is edited or saved by an external text editor or the like (default).

Remark This item is only enabled if the [Enable Rapid Build] check box is selected.

Cautions 1. The rapid build will not finish if this item is selected, and the files to be built have been registered for automatic editing or overwriting (e.g. by commands executed before or after the build).
If the rapid build does not finish, unselect this item, and stop the rapid build.

2. If this item is selected, a file that is registered in the project but does not exist (a file grayed out) will not be observed even if it is registered again by the Explorer etc.
To observe the file, reload the project file, or select this item again after unselecting this item and closing this dialog box.

(3) [Show dependency files in project tree]

☑	Displays the file group on which the source file depends on the project tree.
☐	Does not display the file group on which the source file depends on the project tree (default).

(4) [Output quality report file when build is successful]

☑	Output the quality report file when a build is successful.
☐	Does not output the quality report file when a build is successful (default).

Remarks 1. The quality report file is not output when a rapid build is executed, a debug-dedicated project is built, and compiling or assembling is executed in file units.

2. The following information item is output to the quality report file.
-Time and date on which the file is created
-Log of the build results
-Information on the command file which is used during building
-Information on the detailed version of this product or the current project

3. The quality report file is output with the file name "QuarityReport(*project-name.build-mode-name*).text" to the project folder of each project.
If a file having the same name exists, it will be overwritten.
It is also shown on the project tree, under the Build tool generated files node.

(5) [Enable Break Sound]

☑	Beeps when the execution of a user program is halted due to a break event (hardware or software break).
☐	Does not beep when the execution of a user program is halted due to a break event (hardware or software break) (default).

(6) [Observe downloaded load module files changing]

☑	Observes the load module file downloaded to the debug tool for changes. When there is a change, a message dialog box confirming whether to execute the download will be displayed.
☐	Does not observe the load module file downloaded to the debug tool for changes (default).

(7) [Add source files automatically (for the "Debug Only" project only)]

☑	In a debug-dedicated project, when downloading a load module file into the debug tool, source files are automatically added to the project tree (default).
☐	In a debug-dedicated project, when downloading a load module file into the debug tool, source files are not automatically added to the project tree.

Caution This function is only valid when a load module file has been added to the Download files node of the project tree.
When a load module file has been added in the [Download File Settings] tab on the Property panel of the debug tool, source files are not added to the project tree.

(8) Buttons

Initialize Settings	Returns all currently displayed setting to their default values.

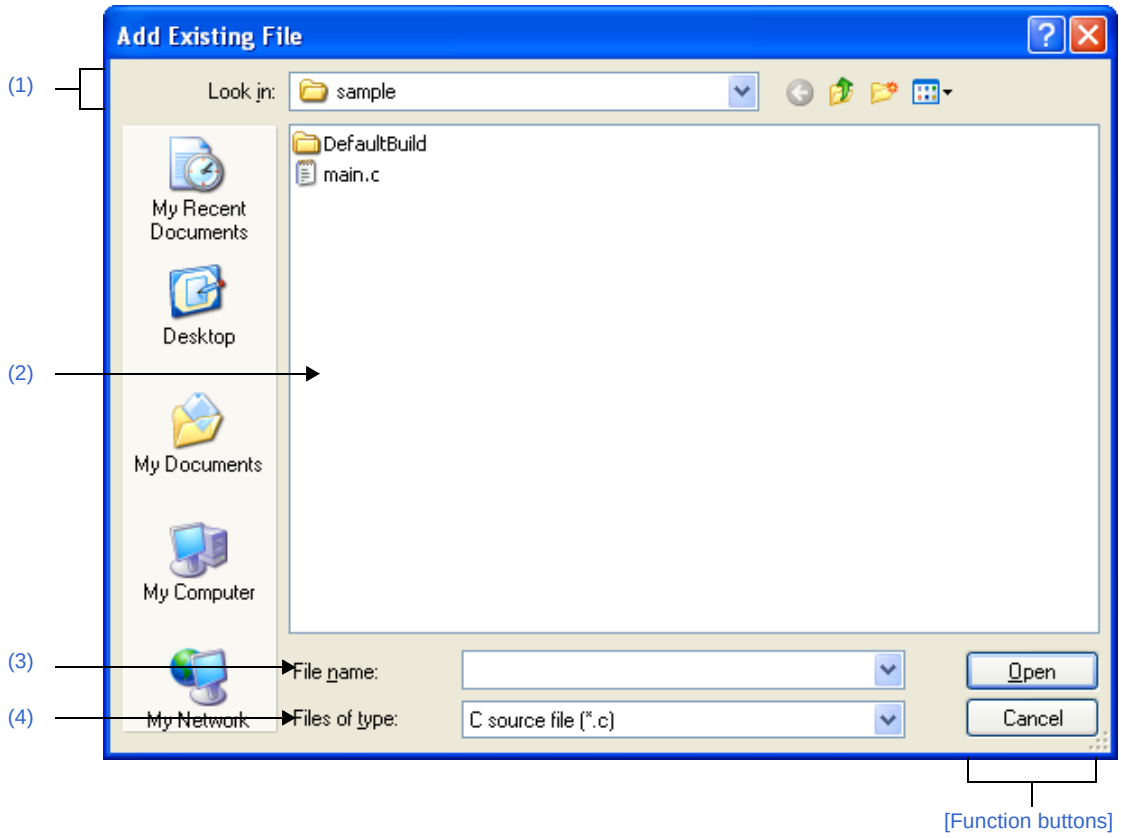
[Function buttons]

Button	Function
Initialize All Settings	Restores all settings on this dialog box to their default values. Note, however, that newly added items in the [General - External Tools] category will not be removed.
OK	Applies all setting and closes this dialog box.
Cancel	Ignores the setting and closes this dialog box.
Apply	Applies all setting (does not close this dialog box).
Help	Displays the help of this dialog box.

Add Existing File dialog box

This dialog box is used to select existing files to add to the project.

Figure A-42. Add Existing File Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- From the [File] menu, select [Add] >> [Add File...].
- On the Project Tree panel, select either one of the Project node, Subproject node, File node, or file, and then select [Add] >> [Add File...] from the context menu.

[Description of each area]

(1) [Look in] area

Select the folder that the file to be added to the project exists.
The project folder is selected by default.

(2) File list area

This area displays the list of the files which match to the selections in the [Look in] and [Files of type].

(3) [File name] area

Designate the name of the file to be added to the project.

(4) [Files of type] area

Designate the type of the file to be added to the project.

C source file (*.c)	C source file (default)
Header file (*.h; *.inc)	Header file
Assembly source file (*.asm; *.s; *.fsy)	Assembly source file
Library file (*.lib)	Library file
Object file (*.obj)	Object file
Relocatable file (*.rel)	Relocatable file
Text file (*.txt)	Text format
All Files (*.*)	All the formats

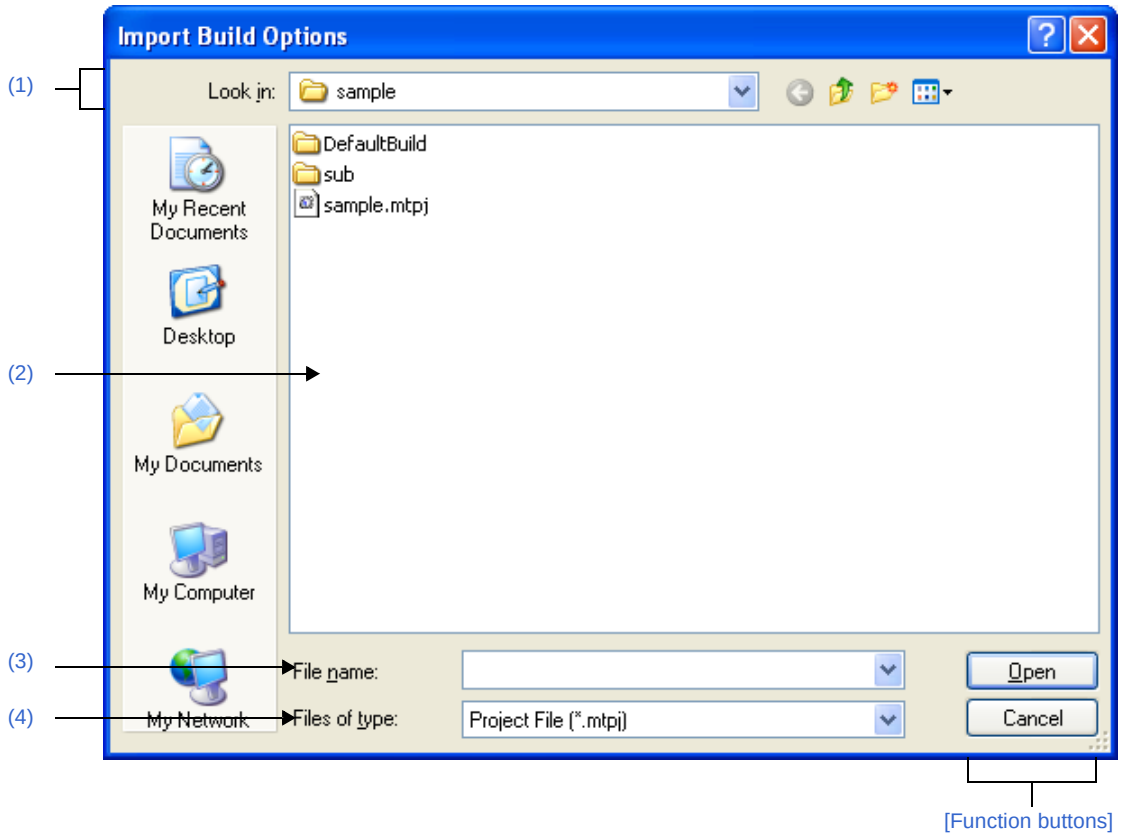
[Function buttons]

Button	Function
Open	Adds the designated file to the project.
Cancel	Closes this dialog box.

Import Build Options dialog box

This dialog box is used to select the target project file for importing the build options.

Figure A-43. Import Build Options Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Project Tree panel, select the Build tool node, and then select [Import Build Options...] from the context menu.

[Description of each area]

(1) [Look in] area

Select the folder that the target project file for importing the build options exists.
The current project folder is selected by default.

(2) File list area

This area displays the list of the files which match to the selections in the [Look in] and [Files of type].

(3) [File name] area

Specify the name of the project file.

(4) [Files of type] area

Select the type of the project file.

Project file (*.mtpj)	Project file
Subproject file (*.mtsp)	Subproject file

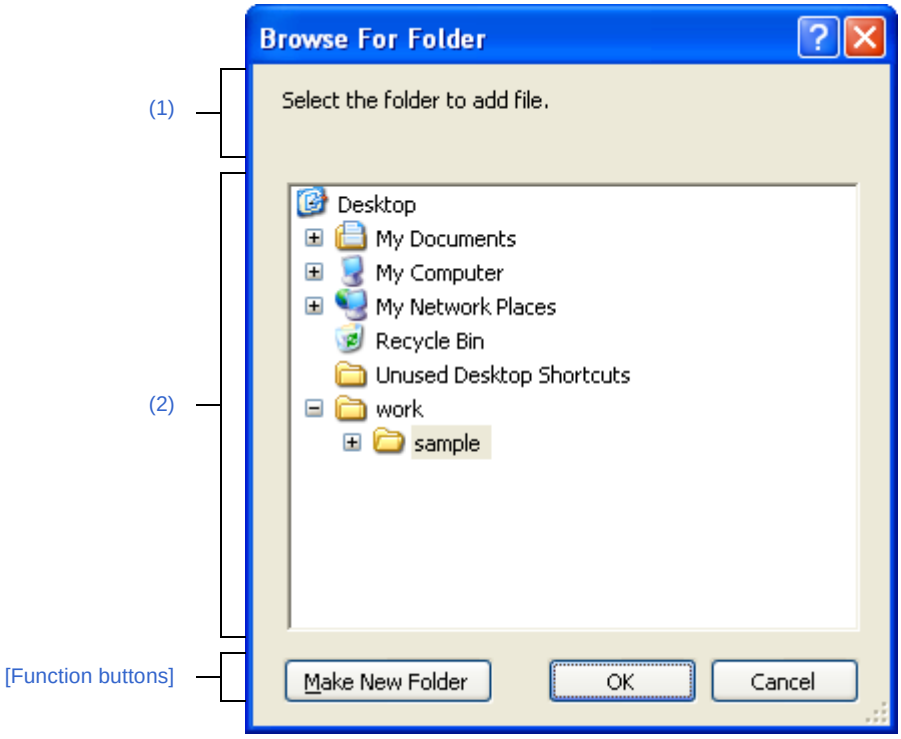
[Function buttons]

Button	Function
Open	Imports the build options of the specified project file to the current project.
Cancel	Closes this dialog box.

Browse For Folder dialog box

This dialog box is used to select the folder and set it to the area that this dialog box is called from.

Figure A-44. Browse For Folder Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- In the Add File dialog box, click the [Refer...] button in the [File location] area.
- In the Path Edit dialog box, click the [Browse...] button in the path edit area.
- On the Property panel, select the following properties, and then click the [...] button.
  - From the [Common Options] tab, [Intermediate file output folder] in the [Output File Type and Path] category, [Output folder] in the [Frequently Used Options(for Link)] category, [Output folder] in the [Frequently Used Options(for Hex Output)] category, [Output folder] in the [Frequently Used Options(for Create Library)] category, and [Error message file output folder] in the [Error Output] category
  - From the [Compile Options] tab, [Output folder for assembly source file] in the [Output File] category and [Output folder for assemble list file] in the [Assemble List] category
  - From the [Assemble Options] tab, [Output folder for assemble list file] in the [Assemble List] category
  - From the [Link Options] tab, [Output folder] in the [Output File] category
  - From the [Hex Output Options] tab, [Output folder], [Output folder for united hex file] in the [Output File] category
  - From the [Create Library Options] tab, [Output folder] in the [Output File] category

- From the [Individual Compile Options] tab, [Output folder for assembly source file] in the [Output File] category, [Output folder for assemble list file] in the [Assemble List] category, and [Error message file output folder] in the [Error Output] category
- From the [Individual Assemble Options] tab, [Output folder for assemble list file] in the [Assemble List] category and [Error message file output folder] in the [Error Output] category

[Description of each area]

(1) Message area

This area displays the message related to the folder to be selected in this dialog box.

(2) Folder location area

Select the folder to be set to the area that this dialog box is called from.
The folder selected by default differs depending on the area that this dialog box is called from.

(a) Add File dialog box

The folder has been set to the area that this dialog box is called from is selected.
When the area has been blank or the path which does not exist has been set, the project folder is selected by default.

(b) Path Edit dialog box and Property panel

The project folder is selected.

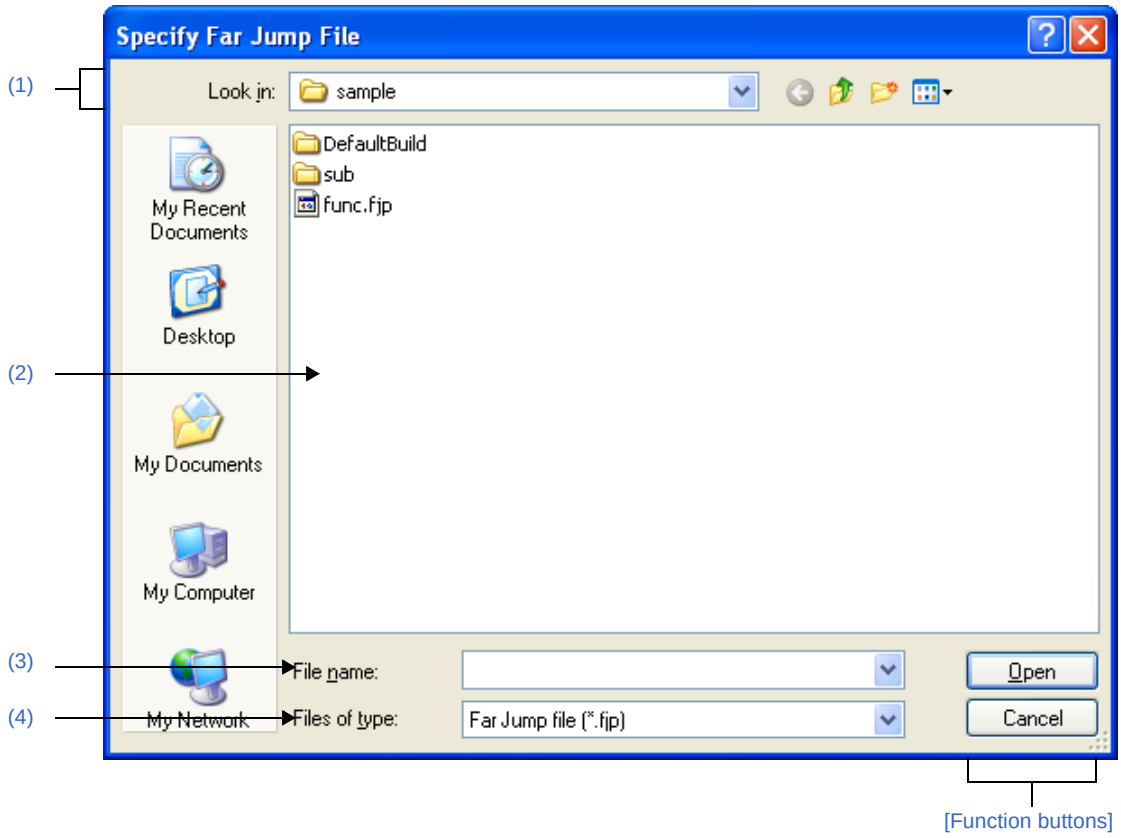
[Function buttons]

Button	Function
Make New Folder	Creates a new folder directly below the root of the selected folder. The default folder name is "New Folder".
OK	Sets the designated folder path to the area that this dialog box is called from.
Cancel	Closes this dialog box.

Specify Far Jump File dialog box

This dialog box is used to select the Far Jump file and set it to the area that this dialog box is called from.

Figure A-45. Specify Far Jump File Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Property panel, select the following properties, and then click the [...] button.
  - From the [Compile Options] tab, [Far Jump file names] in the [Output Code] category
  - From the [Individual Compile Options] tab, [Far Jump file names] in the [Output Code] category

[Description of each area]

(1) [Look in] area

Select the folder where the file to be set to the area that this dialog box is called from exists.
The project folder is selected by default.

(2) File list area

This area displays the list of the files which match to the selections in the [Look in] and [Files of type].

(3) [File name] area

Select the name of the file to be set to the area that this dialog box is called from.

(4) [Files of type] area

Select the type of the file to be set to the area that this dialog box is called from.

Far Jump file (*.fjp)	Far Jump file

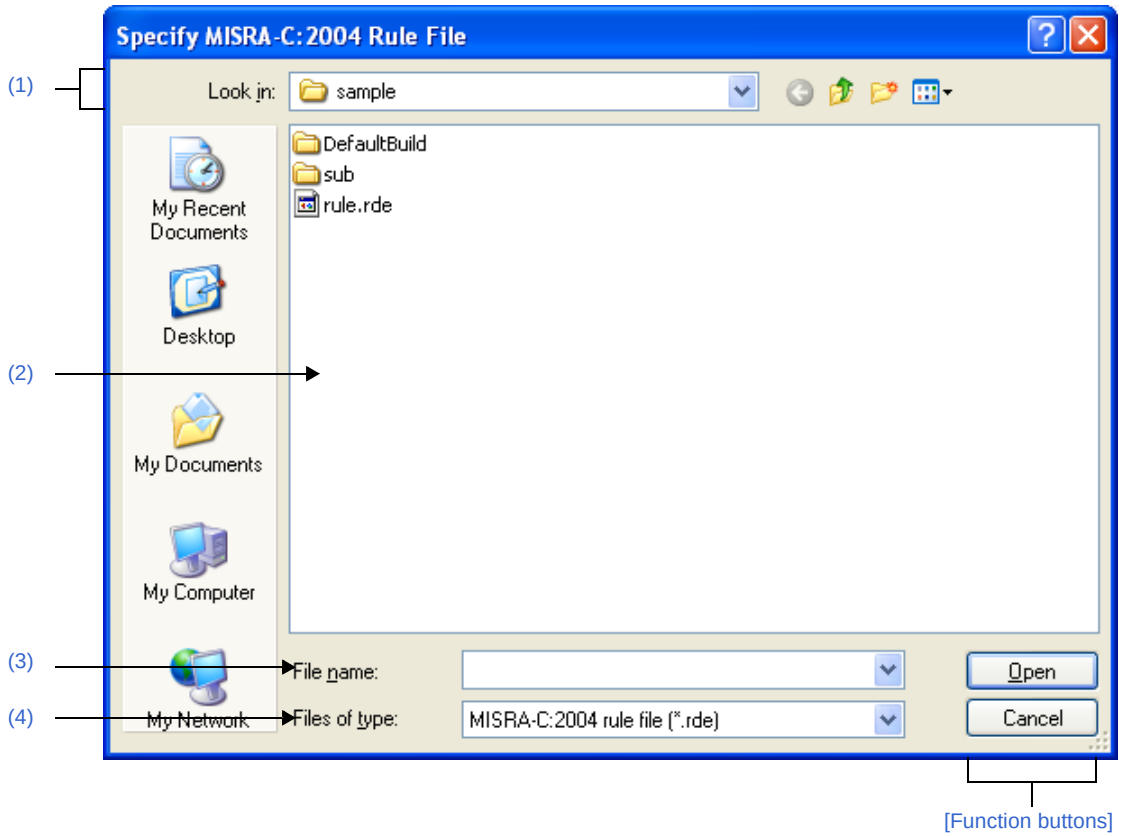
[Function buttons]

Button	Function
Open	Sets the designated file to the area that this dialog box is called from.
Cancel	Closes this dialog box.

Specify MISRA-C:2004 Rule File dialog box

This dialog box is used to select the MISRA-C:2004 rule file and set it to the area that this dialog box is called from.

Figure A-46. Specify MISRA-C:2004 Rule File Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Property panel, select the following properties, and then click the [...] button.
  - From the [Compile Options] tab, [Rule number description file] in the [MISRA-C:2004 Rule Check] category
  - From the [Individual Compile Options] tab, [Rule number description file] in the [MISRA-C:2004 Rule Check] category

[Description of each area]

(1) [Look in] area

Select the folder where the file to be set to the area that this dialog box is called from exists.
The project folder is selected by default.

(2) File list area

File list that matches to the selections in the [Look in] area and [File of type] area is shown.

(3) [File name] area

Select the name of the file to be set to the area that this dialog box is called from.

(4) [Files of type] area

Select the type of the file to be set to the area that this dialog box is called from.

MISRA-C:2004 rule file (*.rde)	MISRA-C:2004 rule file (default)
All files (*.*)	All the formats

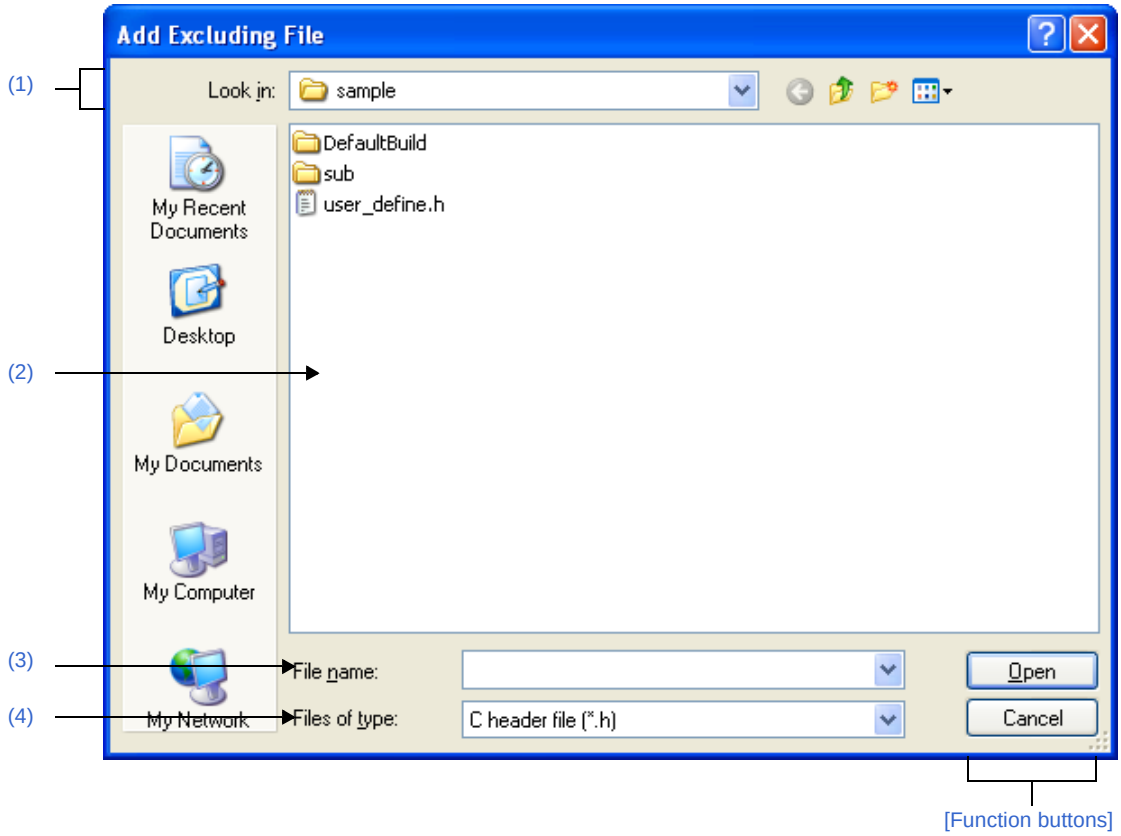
[Function buttons]

Button	Function
Open	Sets the designated file to the area that this dialog box is called from.
Cancel	Closes this dialog box.

Add Excluding File dialog box

This dialog box is used to select the file that will not be checked against the MISRA-C:2004 rules and set it to the area that this dialog box is called from.

Figure A-47. Add Excluding File Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Property panel, after selecting the following properties, open the Path Edit dialog box by clicking the [...] button.
  - From the [Compile Options] tab, [Rule check exclusion file] in the [MISRA-C:2004 Rule Check] category
  - From the [Individual Compile Options] tab, [Rule check exclusion file] in the [MISRA-C:2004 Rule Check] category

  And then click the [Browse...] button in the dialog box.

[Description of each area]

(1) [Look in] area

Select the folder where the file to be set to the area that this dialog box is called from exists.
The project folder is selected by default.

(2) File list area

File list that matches to the selections in the [Look in] area and [File of type] area is shown.

(3) [File name] area

Select the name of the file to be set to the area that this dialog box is called from.

(4) [Files of type] area

Select the type of the file to be set to the area that this dialog box is called from.

C header file (*.h)	C header file (default)
C source file (*.c)	C source file
All files (*.*)	All the formats

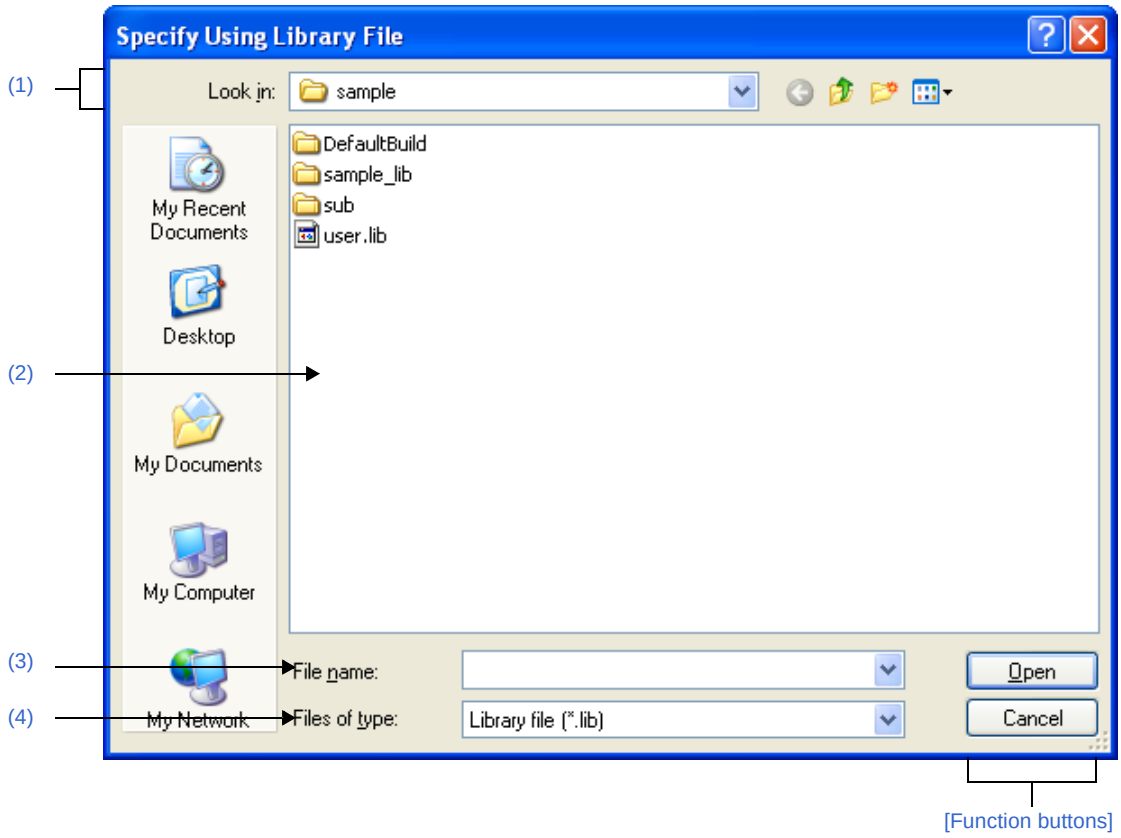
[Function buttons]

Button	Function
Open	Sets the designated file to the area that this dialog box is called from.
Cancel	Closes this dialog box.

Specify Using Library File dialog box

This dialog box is used to select the library file and set it to the area that this dialog box is called from.

Figure A-48. Specify Using Library File Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Property panel, after selecting the following properties, open the Path Edit dialog box by clicking the [...] button.
  - From the [Common Options] tab, [Using libraries] in the [Frequently Used Options(for Link)] category
  - From the [Link Options] tab, [Using libraries] in the [Library] category
  - From the [Create Library Options] tab, [Using libraries] in the [Library] category

  And then click the [Browse...] button in the dialog box.

[Description of each area]

(1) [Look in] area

Select the folder where the file to be set to the area that this dialog box is called from exists.
The project folder is selected by default.

(2) File list area

File list that matches to the selections in the [Look in] area and [File of type] area is shown.

(3) [File name] area

Select the name of the file to be set to the area that this dialog box is called from.

(4) [Files of type] area

Select the type of the file to be set to the area that this dialog box is called from.

Library file (*.lib)	Library file

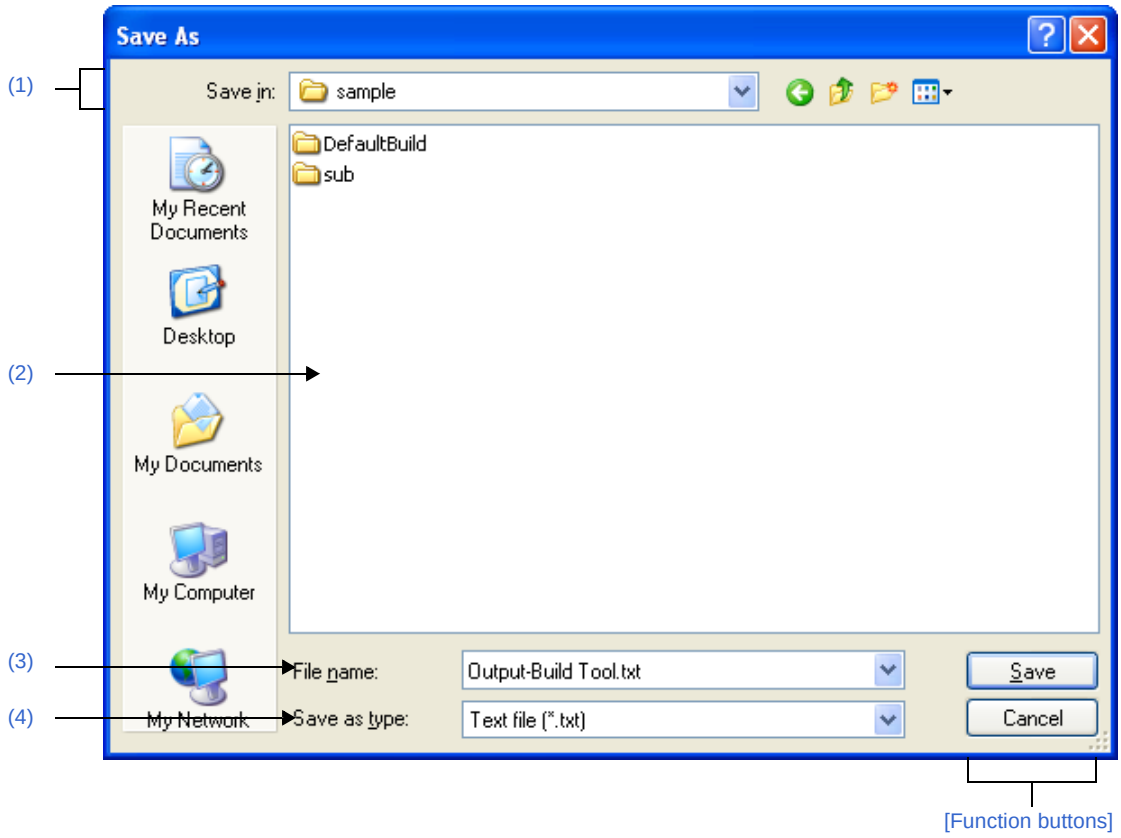
[Function buttons]

Button	Function
Open	Sets the designated file to the area that this dialog box is called from.
Cancel	Closes this dialog box.

Save As dialog box

This dialog box is used to save the editing file or contents of each panel to a file with a name.

Figure A-49. Save As Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- Focus the Editor panel, and then select [Save *file name* As...] from the [File] menu.
- Focus the Output panel, and then select [Save *tab name* As...] from the [File] menu.

[Description of each area]

(1) [Save in] area

Select the folder to save the panel contents in the file.

The following folders are selected by default.

(a) In the Editor panel

The folder that currently editing file is saved is selected.

(b) In the Output panel

The project folder is selected when the file is saved for the first time. The previously selected folder is selected after the second time.

(2) File list area

This area displays the list of the files which match to the selections in the [Save in] area and [Save as type] area.

(3) [File name] area

Designate the name of the file to be saved.

(4) [Save as type] area

(a) In the Editor panel

This area displays any of the following file types depend on the type of the currently editing file.

Remark The following strings are displayed only for the files registered in the project tree.

C source file (*.c)	C source file
Header file (*.h; *.inc)	Header file
Assembly source file (*.asm; *.s)	Assembly source file
Link order specification file (*.mtls)	Link order specification file
Link map file (*.map)	Link map file
Library list file (*.lbp)	Library list file
Intel Hex file (*.hex)	Intel HEX file
Motorola S-record file (*.mot)	Motorola S-record file
Text file (*.txt)	Text format

(b) In the Output panel

This area displays the following file type.

Text file (*.txt)	Text format

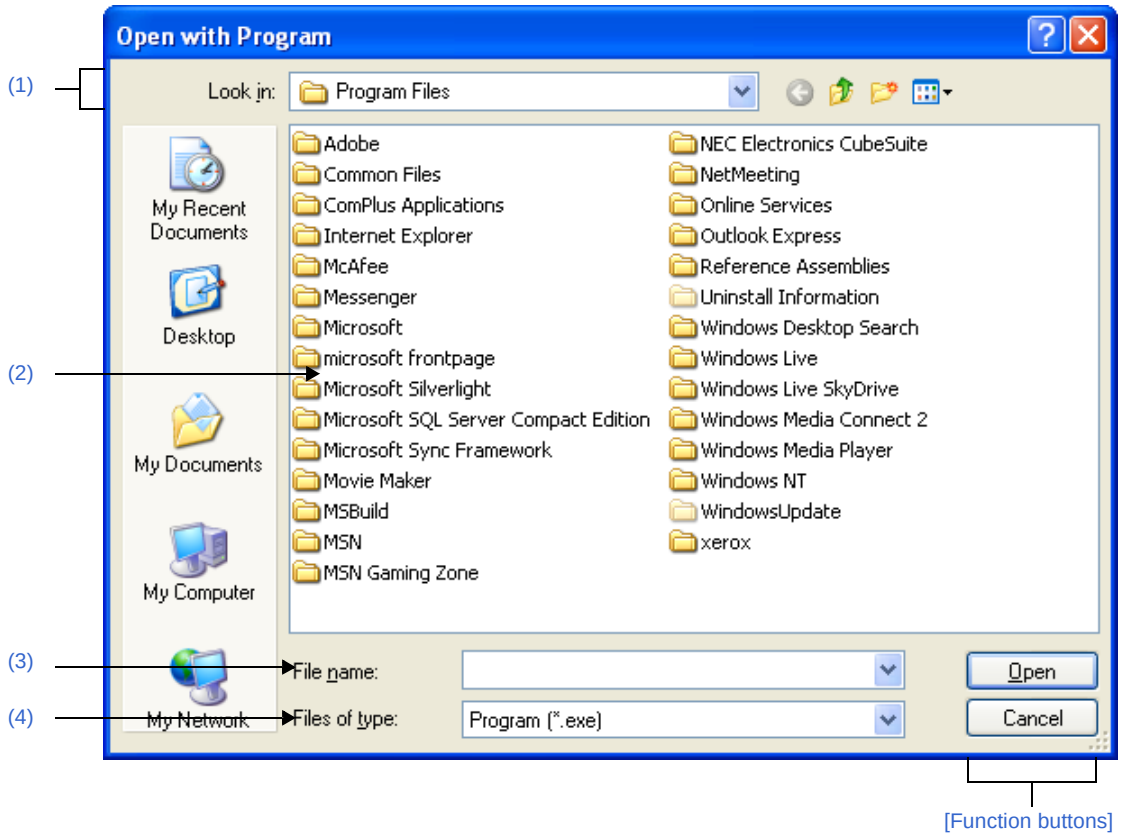
[Function buttons]

Button	Function
Save	Saves the file as the designated name.
Cancel	Closes this dialog box.

Open with Program dialog box

This dialog box is used to select the application to open the file selected in Project Tree.

Figure A-50. Open with Program Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- On the Project Tree panel, select a file, and then select [Open with Selected Application...] from the context menu.

[Description of each area]

(1) [Look in] area

Select the folder where the application to open the file is stored.

Program folder (for Windows XP, "C:\Program Files") is selected by default.

(2) File list area

This area displays the list of the files which match to the selections in the [Look in] and [Files of type].

(3) [File name] area

Specify the executable file name of the application to open the file.

(4) [Files of type] area

Specify the executable file type of the application to open the file.

Program (*.exe)	Executable format (default)
All Files (*.*)	All the formats

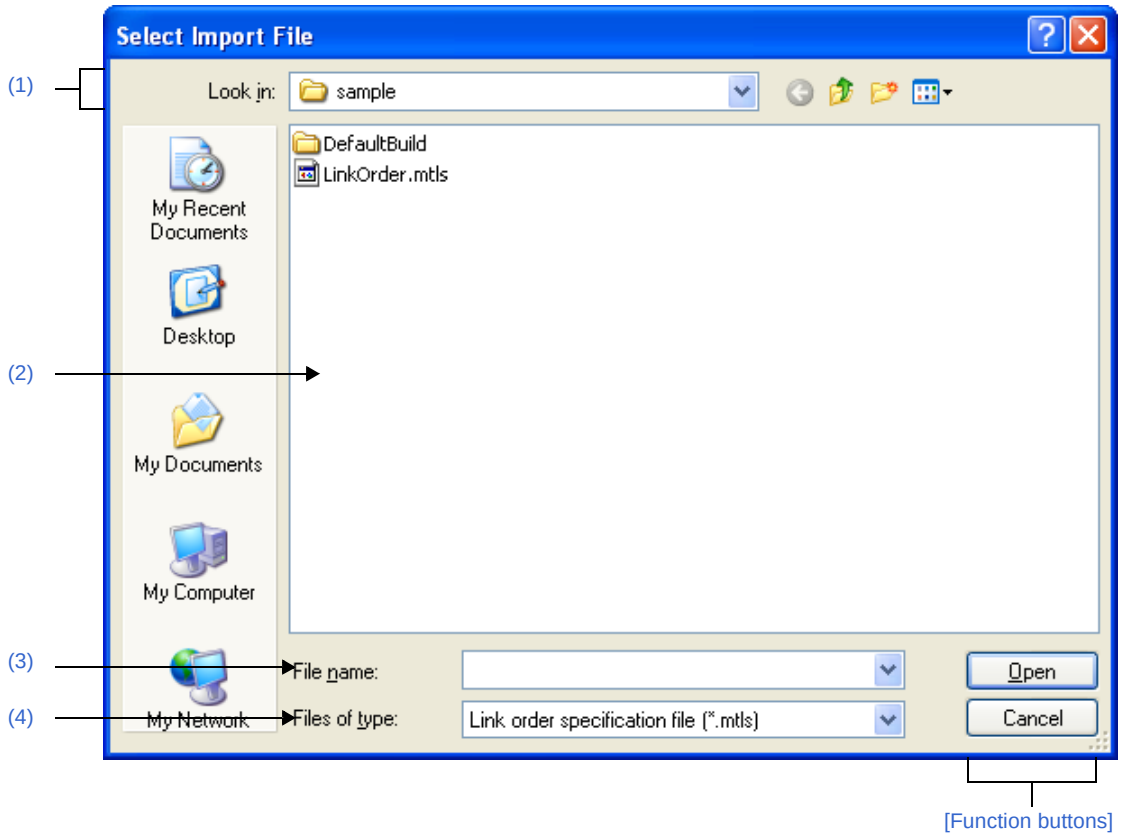
[Function buttons]

Button	Function
Open	Opens the file with the specified application.
Cancel	Closes this dialog box.

Select Import File dialog box

This dialog box is used to select the file and import it to the area that this dialog box is called from.

Figure A-51. Select Import File Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- In the Link Order dialog box, click the [Import] button.
- In the Section Settings dialog box, click the [Import...] button.

[Description of each area]

(1) [Look in] area

Select the folder where the file to be imported to the area that this dialog box is called from exists.
The project folder is selected when the file is selected for the first time. The previously selected folder is selected after the second time.

(2) File list area

This area displays the list of the files which match to the selections in the [Look in] and [Files of type].

(3) [File name] area

Select the name of the file to be imported to the area that this dialog box is called from.

(4) [Files of type] area

Select the type of the file to be imported to the area that this dialog box is called from.

(a) In the Link Order dialog box

Link order specification file (*.mtls)	Link order specification file

(b) In the Section Settings dialog box

Section information file (*.hsi)	Section information file
All Files (*.*)	All the formats

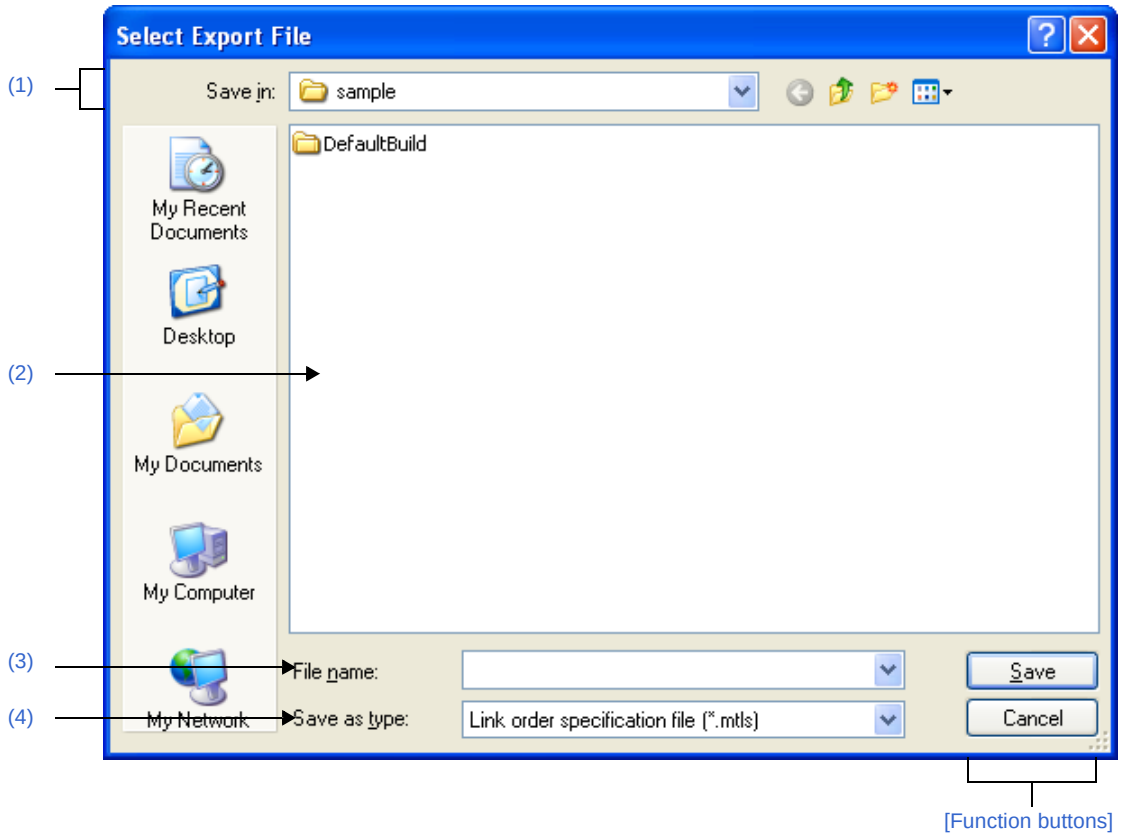
[Function buttons]

Button	Function
Open	Imports the designated file to the area that this dialog box is called from.
Cancel	Closes this dialog box.

Select Export File dialog box

This dialog box is used to output the information of the area that this dialog box is called from.

Figure A-52. Select Export File Dialog Box



The following items are explained here.

- [How to open]
- [Description of each area]
- [Function buttons]

[How to open]

- In the Link Order dialog box, click the [Export] button.
- In the Section Settings dialog box, click the [Export...] button.

[Description of each area]

(1) [Save in] area

Select the folder to store the file to output the information of the area that this dialog box is called from.
The project folder is selected when the file is selected for the first time. The previously selected folder is selected after the second time.

(2) File list area

This area displays the list of the files which match to the selections in the [Save in] area and [Save as type] area.

(3) [File name] area

Specify the name of the output file.

(4) [Save as type] area

This area displays the following file type.

(a) In the Link Order dialog box

Link order specification file (*.mtls)	Link order specification file

(b) In the Section Settings dialog box

Section information file (*.hsi)	Section information file

[Function buttons]

Button	Function
Save	Outputs the file with the designated name.
Cancel	Closes this dialog box.

APPENDIX B COMMAND REFERENCE

This section describes the detailed specifications of each command included in the build tool (CC-RH).

B.1 Overview

CC-RH generates files executable on the target system from source programs described in C language or assembly language.

CC-RH consists of the following commands. A single driver (ccrh) controls all phases from compilation to linking.

ccrh: Compilation driver start command
asrh: Assembler start command
asrh: Optimizing linker start command

Processing of each command is shown below.

(1) Compiler (ccrh)

Performs processing of preprocess directives, comment processing, and optimization for a C source program and then generates an assembly source file.

(2) Assembler (asrh)

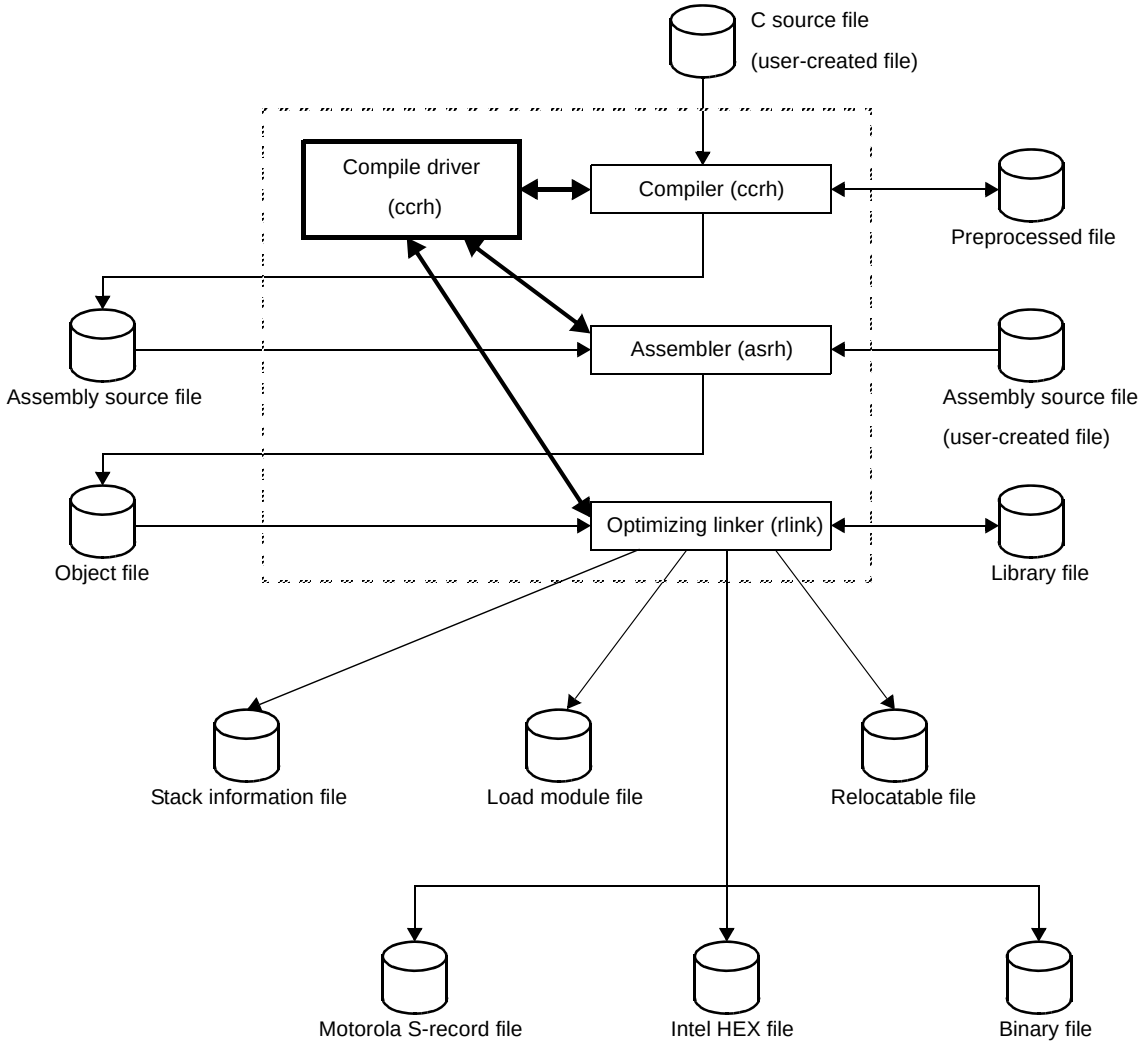
Converts an assembly source program into machine language instructions and then generates a relocatable object file.

(3) Optimizing linker (rlink)

Links an object file and library file, and then generates an executable object file (load module file) on the target system.

It also performs ROMization, performs optimization during linking of relocatable files, creates and edits library files, and converts to Intel HEX files and Motorola S-record files.

Figure B-1. Operation Flow of ccrh



B.2 I/O Files

The I/O files of the ccrh command are shown below.

Table B-1. I/O Files of ccrh Command

File Type	Extension	I/O	Description
C source file	.c	I	Source file described in C language This is created by the user.
Preprocessed file	.i[Note 1]	O	File which the execution result of preprocess processing for the input file is output This is an ASCII image file. This file is output when the -P option is specified.
Assembly source file	.asm[Note 1]	O	Assembly language file generated from C source file by compilation This file is output when the -S option is specified.
	.asm .s	I	Source file described in assembly language This is created by the user.
Header file	free	I	File referred by source files This file is described in C language or assembly language. This is created by the user. The extension is free, but the following is recommended. - #include directive: .h - $include control instruction: .inc
Object file	.obj[Note 1]	I/O	ELF-format file including machine-language information, relocation information relating to machine-language allocation addresses, and symbol information
Assemble list file[Note 2]	.prn[Note 1]	O	List file which has information from the assemble result This file is output when the -Xasm_option=-Xprn_path option is specified.
Library file	.lib[Note 1]	I/O	ELF-format file in which two or more object files are included This file is output when the -Xlk_option=-form=library option is specified.
Load module file	.abs[Note 1]	I/O	ELF-format file of the object code of the link result This is the input file when a hex file is output. This file is output when the -Xlk_option=-form=absolute option is specified. If you specify the -Xlk_option option but not the -form option, the command assumes that the above option has been specified.
Relocatable file	.rel[Note 1]	O	Relocatable object file This file is output when the -Xlk_option=-form=relocate option is specified.
Intel HEX file[Note 2]	.hex[Note 1]	I/O	Load module file converted into the Intel HEX format This file is output when the -Xlk_option=-form=hexadecimal option is specified.
Motorola S-record file[Note 2]	.mot[Note 1]	I/O	Load module file converted into the Motorola S-record This file is output when the -Xlk_option=-form=stype option is specified.

File Type	Extension	I/O	Description
Binary file	.bin[Note 1]	O	Load module file converted into the binary format This file is output when the -Xlk_option=-form=binary option is specified.
Symbol address file	.fsy	I/O	Assembly source file where external defined symbols are described in assembler directives This file is output when the -Xlk_option=-fsymbol option is specified.
Link map file[Note 2]	.map[Note 1]	O	List file which has information from the link result This file is output when the -Xlk_option=-list option is specified.
Library list file[Note 2]	.lbp[Note 1]	O	List file which has information from the library creation result This file is output when the -Xlk_option=-list option is specified.
Stack information file	.sni	O	List file which has information of the stack capacity This file is output when the -Xlk_option=-stack option is specified.
External symbol allocation information file	.bls[Note 1]	O	External variable allocation information file used by the compiler in optimizing access to external variables This file is output when the -Omap option is specified.
Static analysis information file	free	I/O	Information file which this product uses The extension is free, but ".cref" is recommended. This file is output when the -Xcref option is specified.
Error message file	free	O	File which contains error messages The extension is free, but ".err" is recommended. This file is output when the -Xerror_file option is specified.
Subcommand file	free	I	File which contains the parameters of the execution program This is created by the user.

Notes 1. The extension can be changed by specifying the option.

2. See "CHAPTER 3 BUILD OUTPUT LISTS" for details about each file.

B.3 Environment Variable

This section explains the environment variables.

The environment variables of the optimizing linker and the examples when specifying them on the command line are shown below.

- HLNK_LIBRARY1, HLNK_LIBRARY2, HLNK_LIBRARY3

  Specify the default library that the optimizing linker uses.

  The library specified by the -library option has the precedence for linking.

  After that, if unresolved symbols remain, default libraries HLNK_LIBRARY1, HLNK_LIBRARY2, and HLNK_LIBRARY3 are searched in that order.

Example

```
>set HLNK_LIBRARY1=usr1.lib
>set HLNK_LIBRARY2=usr2.lib
>set HLNK_LIBRARY3=usr3.lib
```

- HLNK_TMP

  Specify the folder where the optimizing linker creates temporary files.

  If this environment variable is not specified, the files are created in the current folder.

Example

```
>set HLNK_TMP=D:\workspace\tmp
```

- HLNK_DIR

  Specify the folder where the input files for the optimizing linker are stored.

  The files specified by the -input and -library options are searched from the current folder and the folder specified by HLNK_DIR in that order.

  However, the files specified with wildcard characters are searched in the current folder.

Example

```
>set HLNK_DIR=D:\workspace\obj1;D:\workspace\obj2
```

B.4 Method for Manipulating

This section explains how to manipulate each command.

- Command line operation
- Subcommand file usage
- Set options in CubeSuite+

B.4.1 Command line operation

You can launch the ccrh command (the compilation driver) to perform compilation, assembly, linking, and other actions. The assembler (asrh) and optimizing linker (rlink) can also start by itself.

(1) Specification format

Enter the following on the command line.

```
>ccrh[Δoption]...Δfile[Δfile|Δoption]...
```

```
>asrh[Δoption]...Δfile[Δfile|Δoption]...
```

```
>rlink[{Δfile|Δoption}...]
```

option: Option name
file: File name
[ ]: Can be omitted
...: Pattern in proceeding [ ] can be repeated
{ }: Select from items delimited by the pipe symbol ("|")
Δ: One or more spaces

The following points should be noted when entering a command.

- The specification formats of options depend on the command that is used.
  See "B.5.1 Compile options", "B.5.2 Assemble options"and "B.5.3 Link options" for cautions about options of each command.
- A file name supported by Windows can be specified.
  However, "@" cannot be used at the beginning of a file name because it is regarded as the subcommand file specification.
  "-" cannot be also used at the beginning of a file name because it is regarded as the option specification.
  "(" and ")" cannot be also used for a file name because they are regarded as the part of link options.
- The length that can be specified for a file name depends on Windows (up to 259 characters).
- Uppercase characters and lowercase characters are not distinguished for the alphabet of a file name.
- Two or more files can be specified as input.
  Files which have different types (C source file and assembly source file or object file, and the like) can be mixed.
  Note that two or more files having the same source file name except for the extension cannot be specified (even when they are stored in separate folders).
  In this case, even if there is an error in one file, processing of the remaining files will continue if processing is possible.
  The generated object file is not deleted after linking.

(2) Example of operations

The examples of operations on the command line are shown below.

Remark See "B.5 Option" for details about each option.

(a) Performing compilation, assembly, and linking by one command

C source file "file1.c" is compiled by ccrh, and then assembly source file "file1.asm" is generated.
Next, assembly source file "file1.asm" and "file2.asm" are assembled by asrh, and then object file "file1.obj" and "file2.obj" are generated.
Assemble list file "sample.prn" are also output.
Finally, object file "file1.obj", "file2.obj", and "file3.obj" are linked by rlink, and then link map file "sample.map" and load module file "sample.abs" are generated.

```
>ccrh file1.c file2.asm file3.obj -Xasm_option=-Xprn_path=sample.prn -Xlk_option=-list
-osample.abs -Xcommon=v850e3v5
```

Remark In the ccrh command line, use the -Xasm_option option to specify an option dedicated to asrh; to specify an option dedicated to rlink, use the -Xlk_option option.

(b) Performing compilation and assembly by one command, and linking separately

C source file "file1.c" is compiled by ccrh, and then assembly source file "file1.asm" is generated.
Next, assembly source file "file1.asm" and "file2.asm" are assembled by asrh, and then object file "file1.obj" and "file2.obj" are generated.
Assemble list file "sample.prn" are also output.

```
>ccrh -c file1.c file2.asm -Xasm_option=-Xprn_path=sample.prn -Xcommon=v850e3v5
```

Remark In the ccrh command line, use the -Xasm_option option to specify an option dedicated to asrh.

Object file "file1.obj", "file2.obj", and "file3.obj" are linked by rlink, and then link map file "sample.map" and load module file "sample.abs" are generated.

```
>rlink file1.obj file2.obj file3.obj -output=sample.abs -list
```

(c) Performing compilation, assembly, and linking separately

C source file "file1.c" is compiled by ccrh, and then assembly source file "file1.asm" is generated.

```
>ccrh -S file1.c -Xcommon=v850e3v5
```

Assembly source file "file1.asm" and "file2.asm" are assembled by asrh, and then object file "file1.obj" and "file2.obj" are generated.
Assemble list file "sample.prn" are also output.

```
>asrh -c file1.asm file2.asm -Xprn_path=sample.prn -Xcommon=v850e3v5
```

Object file "file1.obj", "file2.obj", and "file3.obj" are linked by rlink, and then link map file "sample.map" and load module file "sample.abs" are generated.

```
>rlink file1.obj file2.obj file3.obj -output=sample.abs -list
```

B.4.2 Subcommand file usage

A subcommand file is a file that options and file names specified for a command are described.

The command treats the contents of a subcommand file as if they were command-line arguments.

Use a subcommand file when the arguments will not fit on the command line, or when same options are specified repeatedly each time the command is executed.

(1) Using a subcommand file for the compiler and assembler

(a) Cautions about description of a subcommand file

- The arguments to be specified can be coded over several lines.
  However, you cannot start a new line within the name of the option or file.
- A subcommand file cannot be nested.
- The character code contents of a subcommand file cannot be specified by using the -Xcharacter_set option.
  If you use characters other than ASCII in the subcommand file, use the UTF-8 file with BOM.
- The following characters are treated as special characters.
  These special characters themselves are not included in the command line of the ccrh command and deleted.

" (double quotation mark)	The character string until the next double quotation mark is treated as a contiguous character string.
# (sharp)	If this is specified at the beginning of a line, the characters on that line before the end of the line are interpreted as a comment.
^ (circumflex)	The character immediately following this is not treated as a special character.

(b) Example of subcommand file specification

Create subcommand file "sub.txt" using an editor.

```
-Xcommon=v850e3v5
-c
-Dtest
-Idir
-Osize
```

Specify sub.txt by subcommand file specification option "@" on the command line.

```
>ccrh @sub.txt -ofile.obj file.c
```

The command line is expanded as follows.

```
>ccrh -Xcommon=v850e3v5 -c -Dtest -Idir -Osize -ofile.obj file.c
```

(2) Using a subcommand file for the optimizing linker

(a) Cautions about description of a subcommand file

- The leading hyphen ("-") on option names can be omitted.
- A space can be used in place of the equals sign ("=") as the delimiter between the option and parameter.
- Specify one option per one line.
  If the command line cannot fit on a single line, you can use the ampersand ("&") to span multiple lines.
- The -subcommand option cannot be specified in a subcommand file.
- The following characters are treated as special characters.
  These special characters themselves are not included in the command line of the rlink command and deleted.

& (and)	The following line will be treated as a continuation.
; (semicolon)	The characters on that line before the end of the line are interpreted as a comment.

(b) Example of subcommand file specification

Create subcommand file "sub.txt" using an editor.

```
input file2.obj file3.obj       ; This is a comment.
library lib1.lib, &             ; This is a line continued.
lib2.lib
```

Specify sub.txt by subcommand file specification option "-subcommand" on the command line.

```
>rlink file1.obj -subcommand=sub.txt file4.obj
```

The command line is expanded as follows.

```
>rlink file1.obj file2.obj file3.obj -library=lib1.lib,lib2.lib file4.obj
```

B.4.3 Set options in CubeSuite+

This section describes how to set options from CubeSuite+.

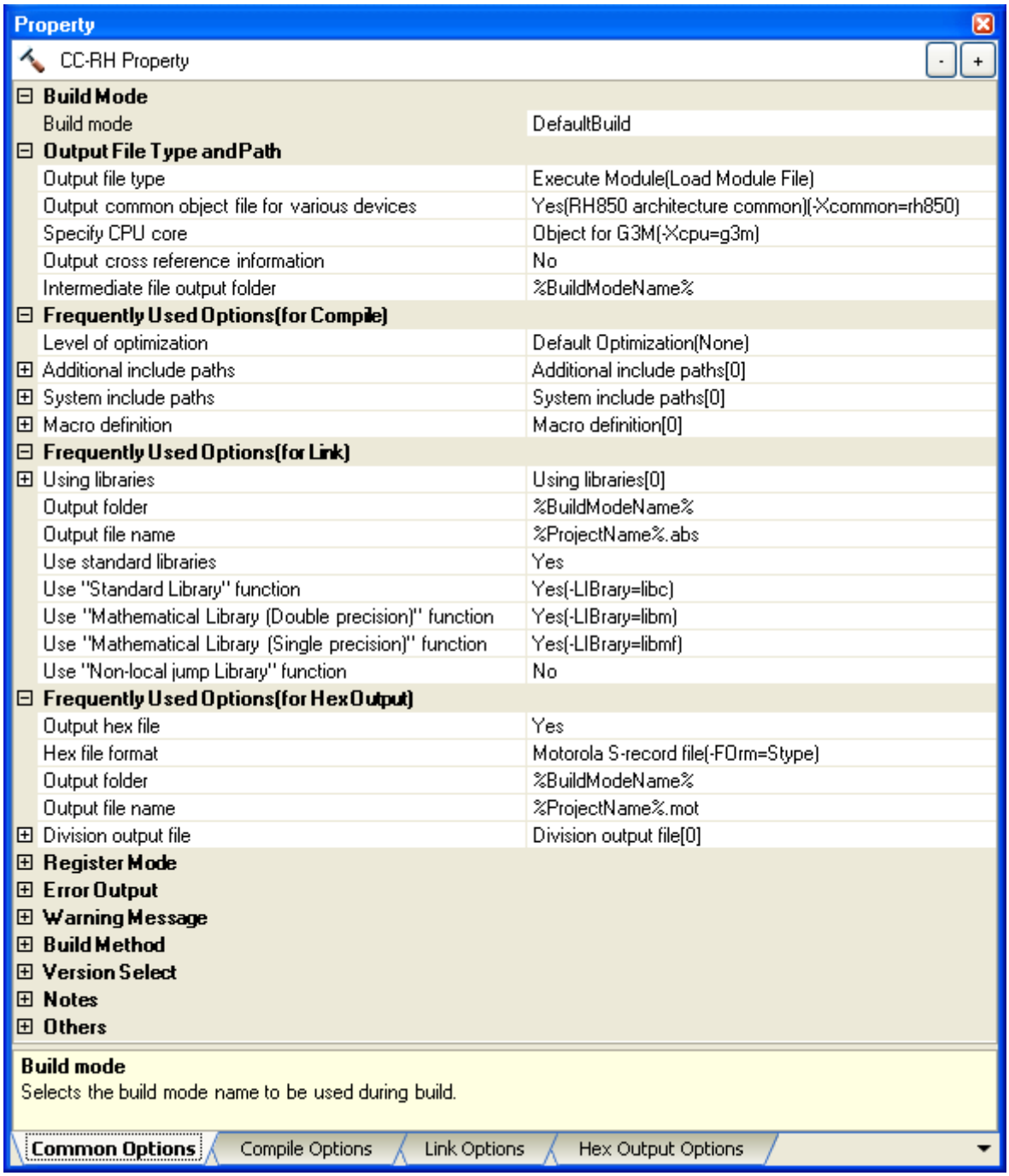
On the CubeSuite+'s Project Tree panel, select the Build Tool node. Next, select [Property] from the [View] menu. The Property panel opens.

- For the application project
  Select the [Common Options] tab, [Compile Options] tab, [Assemble Options] tab, [Link Options] tab, and [Hex Output Options] tab.

- For the library project
  Select the [Common Options] tab, [Compile Options] tab, [Assemble Options] tab, and [Create Library Options] tab.

You can set the various ccrh options by setting the necessary properties in this tab.

Figure B-2. Property Panel



B.5 Option

This section explains ccrh options for each phase.

Compile phase -> See "B.5.1 Compile options"

Assemble phase -> See "B.5.2 Assemble options"

Link phase -> See "B.5.3 Link options"

B.5.1 Compile options

This section explains options for the compile phase.

Caution about options are shown below.

- Uppercase characters and lowercase characters are distinguished for options.
- When numerical values are specified as parameters, decimal or hexadecimal numbers which starts with "0x" ("0X") can be specified.
  Uppercase characters and lowercase characters are not distinguished for the alphabet of hexadecimal numbers.
- When a file name is specified as a parameter, it can include the path (absolute path or relative path).
  When a file name without the path or a relative path is specified, the reference point of the path is the current folder.
- When a parameter includes a space (such as a path name), enclose the parameter in a pair of double quotation marks (").

The types and explanations for options are shown below.

Table B-2. Compile Options

Classification	Option	Description
Version/help display specification	-V	This option displays the version information of ccrh.
	-h	This option displays the descriptions of ccrh options.
Output file specification	-o	This option specifies the output file name.
	-Xobj_path	This option specifies the folder to save an object file generated during compilation.
	-Xasm_path	This option specifies the folder to save an assembly source file generated during compilation.
	-Xprep_path	This option specifies the folder to save the preprocessed file.
Source debugging control	-g	This option outputs information for source debugging.
Device specification	-Xcommon	This option specifies that an object file common to the various devices is generated.
	-Xcpu	This option specifies that an object for the specified core is generated.
Processing interrupt specification	-P	This option executes only preprocess processing for the input file.
	-S	This option does not execute processing after assembling.
	-c	This option does not execute processing after linking.
Preprocessor control	-D	This option defines preprocessor macros and assembler symbols.
	-U	This option deletes the definition of the preprocessor macro or assembler symbol by the -D option.
	-I	This option specifies the folder to search include files.
	-Xpreinclude	This option specifies the file that is included at the top of the compilation unit.
	-Xpreprocess	This option controls outputting the result of preprocessing.

Classification	Option	Description
C language control	-Xansi	This option processes as making C source program comply strictly with the ANSI standard.
	-Xenum_type	This option specifies which integer type the enumeration type handles.
	-Xvolatile	This option specifies external variables as volatile.
	-Xcheck	This option checks the compatibility of a C source file.
	-Xmisra2004	This option checks source code against the MISRA-C: 2004 rules.
	-Xignore_files_misra	This option specifies files that will not be checked against the MISRA-C: 2004 rules.
	-Xcheck_language_extention	This option enables the source-code checking of the MISRA-C:2004 rules, which are partially suppressed by language extensions.
Japanese/Chinese character control	-Xcharacter_set	This option specifies the Japanese/Chinese character code.
Optimization specification	-O	This option specifies the optimization level or the details of each optimization items.
	-Xintermodule	This option performs inter-module optimization.
	-Xinline_strcpy	This option performs inline expansion of standard library functions "strcpy", "strcmp", "memcpy", and "memset" calls.
	-Xmerge_string	This option merges string literals.
	-Xalias	This option performs optimization with consideration for the type of the data indicated by the pointer.
	-Xmerge_files	This option merges two or more C source files and compiles them.
	-Xwhole_program	This option performs optimization assuming that the files to be compiled comprise the entire program.

Classification	Option	Description
Generated code control	-Xpack	This option performs the structure packing.
	-Xbit_order	This option specifies the order of bit-field members.
	-Xpass_source	This option outputs a C source program as a comment to the assembly source file.
	-Xswitch	This option specifies a mode in which the code of a switch statement is to be output.
	-Xreg_mode	This option specifies the register mode.
	-Xreserve_r2	This option reserves the r2 register.
	-Xep	This option specifies how to handle the ep register.
	-Xfloat	This option controls generating floating-point calculation instructions.
	-Xcall_jump	This option controls generating function-call branch instructions.
	-Xfar_jump	This option controls outputting far jump.
	-Xdiv	This option generates the div and divu instructions for division.
	-Xcheck_div_ov	This option checks the OV flag at division.
	-Xuse_fmaf	This option generates product-sum operation instructions.
	-Xunordered_cmpf	This option detects invalid operation exceptions in floating-point comparison.
	-Xmulti_level	This option specifies the generation of a multi-core program.
	-Xpatch	This option applies a patch.
Information file output control	-Xcref	This option outputs the static analysis information file.
Error output control	-Xerror_file	This option outputs error messages to a file.
Warning message output control	-Xno_warning	This option suppresses outputting warning messages of the specified number.
Phase individual option specification	-Xasm_option	This option specifies assemble options.
	-Xlk_option	This option specifies link options.
Subcommand file specification	@	This option specifies a subcommand file.

Version/help display specification

The version/help display specification options are as follows.

- -V
- -h

-V

This option displays the version information of ccrh.

[Specification format]

```
-V
```

- Interpretation when omitted
  Compilation is performed without displaying the version information of ccrh.

[Detailed description]

- This option outputs the version information of ccrh to the standard error output.
  It does not execute compilation.

[Example of use]

- To output the version information of ccrh to the standard error output, describe as:

```
>ccrh -V -Xcommon=v850e3v5
```

-h

This option displays the descriptions of ccrh options.

[Specification format]

```
-h
```

- Interpretation when omitted
  The descriptions of ccrh options are not displayed.

[Detailed description]

- This option outputs the descriptions of ccrh options to the standard error output.
  It does not execute compilation.

[Example of use]

- To output the descriptions of ccrh options to the standard error output, describe as:

```
>ccrh -h -Xcommon=v850e3v5
```

Output file specification

The output file specification options are as follows.

- -o
- -Xobj_path
- -Xasm_path
- -Xprep_path

-o

This option specifies the output file name.

[Specification format]

```
-ofile
```

- Interpretation when omitted

The output file name differs depending on the specificated option.

The file is output to the current folder.

- When the -P option is specified
  The output file name will be the input file name with the extension replaced by ".i".
- When the -S option is specified
  The output assembly source file name will be the source file name with the extension replaced by ".asm".
- When the -c option is specified
  The output object file name will be the source file name with the extension replaced by ".obj".
- Other than above
  The output load module file name will be the first input file name with the extension replaced by ".abs".

[Detailed description]

- This option specifies the output file name as *file*.
- If *file* already exists, it will be overwritten.
- This option is valid when processing is interrupted by specifying the -P, -S, or -c option.
  - If this option is specified with the -P option
    It is assumed that is the name of the file containing the results of preprocessing performed on the input file has been specified as *file*.
  - If this option is specified with the -S option
    It is assumed that an assembly source file name has been specified as *file*.
  - If this option is specified with the -c option
    It is assumed that an object file name has been specified as *file*.
  - Other than above
    It is assumed that a load module file name has been specified as *file*.
- An error will occur if two or more files are output.
- An error will occur if *file* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Individual Compile Options] tab, [Object file name] in the [Output File] category

[Example of use]

- To output the load module file with "sample.abs" as the file name, describe as:

```
>ccrh -osample.abs -Xcommon=v850e3v5 main.c
```

-Xobj_path

This option specifies the folder to save an object file generated during compilation.

[Specification format]

```
-Xobj_path[=path]
```

- Interpretation when omitted
  The object file is saved under the source file name with the extension replaced by ".obj" to the current folder.

[Detailed description]

- This option specifies the folder to save an object file generated during compilation as *path*.
- If an existing folder is specified as *path*, the object file is saved under the source file name with the extension replaced by ".obj" to *path*.
  An error will occur if a nonexistent folder is specified.
- An existing file can be specified as path.
  If one object file is output, it will be saved with *path* as the file name.
  If two or more object files are output, an error will occur.
  An error will occur if a nonexistent file is specified.
- If "=*path*" is omitted, the object file is saved under the C source file name with the extension replaced by ".obj".
- If two or more files with the same name (even if they are in different folders) are specified as source files, then a warning is output, and an object file is only saved for the last source file to be specified.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Intermediate file output folder] in the [Output File Type and Path] category

[Example of use]

- To save the object file generated during compilation to foler "D:\sample", describe as:

```
>ccrh -Xobj_path=D:\sample -Xcommon=v850e3v5 main.c
```

-Xasm_path

This option specifies the folder to save an assembly source file generated during compilation.

[Specification format]

```
-Xasm_path[=path]
```

- Interpretation when omitted
  An assembly source file will not be output (except when specifying the -S option).

[Detailed description]

- This option specifies the folder to save an assembly source file generated during compilation as *path*.
- If an existing folder is specified as *path*, the assembly source file is saved under the C source file name with the extension replaced by ".asm" to *path*.
  An error will occur if a nonexistent folder is specified.
- An existing file can be specified as *path*.
  If one assembly source file is output, it will be saved with *path* as the file name.
  If two or more assembly source files are output, an error will occur.
  An error will occur if a nonexistent file is specified.
- If "=*path*" is omitted, the assembly source file is saved under the C source file name with the extension replaced by ".asm".
- If two or more files with the same name (even if they are in different folders) are specified as source files, then a warning is output, and an assembly source file is only saved for the last source file to be specified.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Output assembly source file], [Output folder for assembly source file] in the [Output File] category
  - From the [Individual Compile Options] tab, [Output assembly source file], [Output folder for assembly source file] in the [Output File] category

[Example of use]

- To save the assembly source file generated during compilation to foler "D:\sample", describe as:

```
>ccrh -Xasm_path=D:\sample -Xcommon=v850e3v5 main.c
```

-Xprep_path

This option specifies the folder to save the preprocessed file.

[Specification format]

```
-Xprep_path[=path]
```

- Interpretation when omitted
  A preprocessed file will not be output (except when specifying the -P option).

[Detailed description]

- This option specifies the folder to save a preprocessed file as *path*.
- If an existing folder is specified as *path*, the preprocessed file is saved under the C source file name with the extension replaced by ".i" to *path*.
  An error will occur if a nonexistent folder is specified.
- An existing file can be specified as *path*.
  If one preprocessed file is output, it will be saved with *path* as the file name.
  If two or more preprocessed files are output, an error will occur.
  An error will occur if a nonexistent file is specified.
- If two or more files with the same name (even if they are in different folders) are specified as source files, then a warning is output, and a preprocessed file is only saved for the last source file to be specified.
- If two or more files with the same name (even if they are in different folders) are specified as source files, then a warning is output, and a preprocessed file is only saved for the last source file to be specified.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Output folder for preprocessed source file] in the [Output File] category
  - From the [Individual Compile Options] tab, [Output folder for preprocessed source file] in the [Output File] category

[Example of use]

- To save the preprocessed file to folder "D:\sample", describe as:

```
>ccrh -Xprep_path=D:\sample -Xcommon=v850e3v5 main.c
```

Source debugging control

The source debugging control option is as follows.

- -g

-g

This option outputs information for source debugging.

[Specification format]

```
-g
```

- Interpretation when omitted
  Information for source debugging will not be output.

[Detailed description]

- This option outputs information for source debugging to the output file.
- Source debugging can be performed by specifying this option.
- If this option and an optimization option are specified at the same time, the ease of debugging could be affected.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Add debug information] in the [Debug Information] category
  - From the [Individual Compile Options] tab, [Add debug information] in the [Debug Information] category

[Example of use]

- To output information for source debugging to the output file, describe as:

```
>ccrh -g -Xcommon=v850e3v5 main.c
```

Device specification

The device specification option is as follows.

- -Xcommon
- -Xcpu

-Xcommon

This option specifies that an object file common to the various devices is generated.

[Specification format]

```
-Xcommon=series
```

- Interpretation when omitted
  An error will occur.
  However, an error will not occur if the -V, -h, or -P option is specified.

[Detailed description]

- This option specifies that an object file common to the various devices is generated.
- When this option is specified, only instructions are used in the instruction set architecture of the target, and common magic number *series* that supports the instruction set architecture is embed into the object file.
- The item that can be specified as *series* is shown below.
  An abort error will occur if any other item is specified.

v850e3v5	It is possible to link to the model with the V850E3V5 instruction set architecture, which is specified as a target device.
rh850	The result is the same as when "v850e3v5" is specified.

- An error will occur if *series* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Output common object file for various devices] in the [Output File Type and Path] category

[Example of use]

- To embed the magic number common to models with the instruction set architectures superior to V850E3V5 into the object file to be generated, describe as:

```
>ccrh -Xcommon=v850e3v5 -c main.c
```

-Xcpu

This option specifies that an object for the specified core is generated.

[Specification format]

```
-Xcpu=core
```

- Interpretation when omitted
  If the -Xcommon option is specified, an object for G3M is generated.

[Detailed description]

- This option specifies that an object for core *core* is generated.
- The items that can be specified as *core* are shown below.
  An abort error will occur if any other item is specified.

g3m	Generates an object for G3M.
g3k	Generates an object for G3K.

- An error will occur if *core* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Specify CPU core] in the [Output File Type and Path] category

[Example of use]

- To generate an object for G3M, describe as:

```
>ccrh -Xcpu=g3m -c -Xcommon=v850e3v5 main.c
```

Processing interrupt specification

The processing interrupt specification options are as follows.

- -P
- -S
- -c

-P

This option executes only preprocessing for the input file.

[Specification format]

```
-P
```

- Interpretation when omitted
  Processing is continued after preprocessing.
  The preprocessed file are not output.

[Detailed description]

- This option executes only preprocessing for the input file and outputs the results to a file.
- The output file name will be the input file name with the extension replaced by ".i".
- The output file name can be specified by specifying this option and the -o option.
- The contents of the output file can be controlled by specifying the -Xpreprocess option.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Output assembly source file] in the [Output File] category
  - From the [Individual Compile Options] tab, [Output preprocessed source file] in the [Output File] category

[Example of use]

- To execute only preprocessing for the input file and output the results to file "main.i", describe as:

```
>ccrh -P -Xcommon=v850e3v5 main.c
```

-S

This option does not execute processing after assembling.

[Specification format]

```
-S
```

- Interpretation when omitted
  Processing is continued after assembling.

[Detailed description]

- This option does not execute processing after assembling.
- The assembly source file is output under the source file name with the extension replaced by ".asm".
- The output file name can be specified by specifying this option and the -o option.

[Example of use]

- To output assembly source file "main.asm" without executing any processing after the assembling, describe as:

```
>ccrh -S -Xcommon=v850e3v5 main.c
```

-c

This option does not execute processing after linking.

[Specification format]

```
-c
```

- Interpretation when omitted
  Processing is continued after linking.

[Detailed description]

- This option does not execute processing after linking.
- The object file is output under the source file name with the extension replaced by ".obj".
- The output file name can be specified by specifying this option and the -o option.

[Example of use]

- To output object file "main.obj" without executing any processing after the linking, describe as:

```
>ccrh -c -Xcommon=v850e3v5 main.c
```

Preprocessor control

The preprocessor control options are as follows.

- -D
- -U
- -I
- -Xpreinclude
- -Xpreprocess

-D

This option defines preprocessor macros and assembler symbols.

[Specification format]

```
-Dname[=def][name[=def]]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option defines *name* as a preprocessor macro or user-defined symbol of the assembler.
- This is equivalent to adding "#define *name def*" or "SET *name def*" (only assembly source program) at the beginning of the source program.
- If *name* contains characters that are allowed in an assembler symbol, but which cannot be used in a preprocessor macro ("@", ".", and "~"), a warning will be output, and it is defined as an assembler symbol only.
- This option can be used to redefine C language macros that have been defined already: __LINE__, __FILE__, __DATE__, __TIME__, and __CCRH__ (except for -D__CCRH__[=1]).
  An error will occur if these are redefined when the input file is a C source file.
- An error will occur if *name* is omitted.
- If "=*def*" is omitted, *def* is regarded as 1.
- This option can be specified more than once.
- If both this option and -U option are specified for the same preprocessor macro and assembler symbol, the option specified last will be valid.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Macro definition] in the [Frequently Used Options(for Compile)] category
  - From the [Compile Options] tab, [Macro definition] in the [Preprocess] category
  - From the [Individual Compile Options] tab, [Macro definition] in the [Preprocess] category

[Example of use]

- To define "sample=256" as a preprocessor macro, describe as:

```
>ccrh -Dsample=256 -Xcommon=v850e3v5 main.c
```

-U

This option deletes the definition of the preprocessor macro or assembler symbol by the -D option.

[Specification format]

```
-Uname[,name]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option deletes the definition of the preprocessor macro or user-defined symbol of the assembler *name* by the -D option.
- This is equivalent to adding "#undef *name*" at the beginning of the source program.
- An error will occur if *name* is omitted.
- This option cannot delete the definition by describing "#define *name def*" and ".SET *name def*" (only assembly source program).
- This option can be used to undefine C language macros that have been defined already, but it cannot undefine the following macros: __LINE__, __FILE__, __DATE__, __TIME__, __CCRH__, or __CCRH.
  An error will occur if these are specified for *name*.
- This option can be specified more than once.
- If both this option and -D option are specified for the same preprocessor macro and assembler symbol, the option specified last will be valid.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Macro definition] in the [Preprocess] category
  - From the [Individual Compile Options] tab, [Macro definition] in the [Preprocess] category

[Example of use]

- To delete the definition of preprocessor macro "test" by the -D option, describe as:

```
>ccrh -Utest -Xcommon=v850e3v5 main.c
```

-I

This option specifies the folder to search include files.

[Specification format]

```
-Ipath[,path]...
```

- Interpretation when omitted
  The include file is searched from the standard include file folder.

[Detailed description]

- This option specifies the folder to search include files which are read by preprocessor directive "#include"or assembler control instruction "$INCLUDE/$BINCLUDE" as *path*.
  Include files are searched according to the following sequence.

(1) #include

(a) Folder with source files (When files are specified by using " ")

(b) Path specified by the -I option

(c) Standard include file folder[Note]

Note *product install folder*\CubeSuite+\ccrh\V*x.xx*\inc

(2) $INCLUDE/$BINCLUDE

(a) Path specified by the -I option

(b) Folder with source file

(c) Folder of original C source file

(d) Current folder

- If *path* does not exist, a warning will be output.
- An error will occur if *path* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Additional include paths], [System include paths] in the [Frequently Used Options(for Compile)] category
  - From the [Compile Options] tab, [Additional include paths], [System include paths] in the [Preprocess] category
  - From the [Individual Compile Options] tab, [Additional include paths], [Use whole include paths specified for build tool] in the [Preprocess] category

[Example of use]

- To search include files from the current folder, folder D:\include, the standard folder in that order, describe as:

```
>ccrh -ID:\include -Xcommon=v850e3v5 main.c
```

-Xpreinclude

This option specifies the file that is included at the top of the compilation unit.

[Specification format]

```
-Xpreinclude=file[,file]
```

- Interpretation when omitted
  It is assumed that the file that is included at the top of the compilation unit does not exist.

[Detailed description]

- This option specifies the file that is included at the top of the compilation unit as *file*.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Include files at head of compiling units] in the [Preprocess] category
  - From the [Individual Compile Options] tab, [Include files at head of compiling units] in the [Preprocess] category

[Example of use]

- To include file "sample.h" at the top of the compilation unit, describe as:

```
>ccrh main.c -Xpreinclude=sample.h -Xcommon=v850e3v5
```

-Xpreprocess

This option controls outputting the result of preprocessing.

[Specification format]

```
-Xpreprocess=string[,string]
```

- Interpretation when omitted
  The comments and line number information of the C source are not output to the preprocessed file.

[Detailed description]

- This option outputs the comments and line number information of the C source to the preprocessed file.
- This option is valid only when the -P option is specified.
  If the -P option is not specified, this option will be ignored.
- The items that can be specified as *string* are shown below.
  An error will occur if any other item is specified.

comment	Outputs the comments of the C source.
line	Outputs line number information[Note].

Note The format of the line number information is shown below.

```
#line line-number "file-name"
```

- *line-number* is a decimal number, and the maximum value is the maximum number of unsigned int.
- In the full path of *file-name*, "\\" is converted to "\", and "" to '\"'.
  Other than printable characters (including spaces) are output as "*\3-digit octal number* " (e.g. "\\%03o").
  Line feed characters are converted to "\\n".
- If an input source file contains the preprocessor directive '#*number* "*string*"' or '#line *number* "*string*"', then *number* is used as *line-number*, and *string* as *file-name*.

- An error will occur if *string* is omitted.
- It is output in the standard character encoding of the OS.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Output C source comments to preprocessed file], [Output line number information to preprocessed file] in the [Preprocess] category
  - From the [Individual Compile Options] tab, [Output C source comments to preprocessed file], [Output line number information to preprocessed file] in the [Preprocess] category

[Example of use]

- To output the comments and line number information of the C source to the preprocessed file, describe as:

```
>ccrh -Xpreprocess=comment,line -P -Xcommon=v850e3v5 main.c
```

The following example is equivalent to the example above.

```
>ccrh -Xpreprocess=comment -Xpreprocess=line -P -Xcommon=v850e3v5 main.c
```

C language control

The C language control options are as follows.

- -Xansi
- -Xenum_type
- -Xvolatile
- -Xcheck
- -Xmisra2004
- -Xignore_files_misra
- -Xcheck_language_extention

-Xansi

This option processes as making C source program comply strictly with the ANSI standard.

[Specification format]

```
-Xansi
```

- Interpretation when omitted
  Compatibility with the conventional C language specifications is conferred and processing continues after warning is output.

[Detailed description]

- This option processes as making C source program comply strictly with the ANSI standard[Note] and outputs an error or warning for a specification that violates the standard.
- When this option is specified, macro name "__STDC__" is defined as the macro that the value is 1.
- Processing when compiling in strict adherence to the language specification is as follows.
  - Bit fields
    An error will occur if a type other than an int type is specified in a bit field.
    If this option is not specified, specifying a type other than an int type will be enabled (A warning will not be output).
  - #line-number
    An error will occur.
    If this option is not specified, "#line-number" will be handled in the same way as "#line line-number".
  - Argument of function for which #pragma inline is specified
    If the type of the return value or parameter is different but type conversion is possible between the specified function call and definition, an error will occur.
    If this option is not specified, the type of the return value is converted to the type at the call side, the parameters are converted to the type of the function definition, and inline expansion is performed.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Compile strictly according to ANSI standards] in the [C Language] category
  - From the [Individual Compile Options] tab, [Compile strictly according to ANSI standards] in the [C Language] category

Note This is a standard specified by ISO/IE C9899:1990 (C90).
Although ccrh also accepts some of the specifications added by ISO/IE C9899:1999 (C99), if this option is specified, code in violation of the standard will cause an error.

[Example of use]

- To process as making C source program comply strictly with the ANSI standard and output an error or warning for a specification that violates the standard, describe as:

```
>ccrh -Xansi -Xcommon=v850e3v5 main.c
```

-Xenum_type

This option specifies which integer type the enumeration type handles.

[Specification format]

```
-Xenum_type=string
```

- Interpretation when omitted
  The enumeration type is handled as signed int.

[Detailed description]

- This option specifies which integer type the enumeration type handles.
- The items that can be specified as *string* are shown below.
  An error will occur if any other item is specified.

auto	Each enumerated type is handled as the smallest integer type capable of expressing all the enumerators in that type.

- An error will occur if *string* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Enumeration type] in the [C Language] category

[Example of use]

- To handle each enumerated type as the smallest integer type capable of expressing all the enumerators in that type, describe as:

```
>ccrh -Xenum_type=auto -Xcommon=v850e3v5 main.c
```

-Xvolatile

This option specifies external variables as volatile.

[Specification format]

```
-Xvolatile
```

- Interpretation when omitted
  Only the volatile-qualified variables are handled as if they were volatile-declared.

[Detailed description]

- This option handles all external variables as if they were volatile-declared.
  The access count and access order for external variables are exactly the same as those described in the C source file.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Handle external variables as if they are volatile qualified] in the [C Language] category
  - From the [Individual Compile Options] tab, [Handle external variables as if they are volatile qualified] in the [C Language] category

[Example of use]

- To handle all external variables as if they were volatile-declared, describe as:

```
>ccrh -Xvolatile -Xcommon=v850e3v5 main.c
```

-Xcheck

This option checks the compatibility of a C source file.

[Specification format]

-Xcheck=*comp*

- Interpretation when omitted
  The compatibility of a C source file is not checked.

[Detailed description]

- This option checks the C source file coded for the compiler specified as *comp*. It checks for option specifications and source code that will impact compatibility when compiled with this compiler, and outputs warnings or errors about any impacts found.
- The items that can be specified as *comp* are shown below.
  An error will occur if any other item is specified.

shc	Checks the C source file that has been coded for the SH compiler.

- An error will occur if *comp* is omitted.
- The main check items are shown below.
  - Options: -Xbit_order=*pos*
    The settings which are not defined in the language specification and depend on implementation differ in each compiler.
    Confirm the selections of the options which were output in the message.
  - Extended functions: #pragma section, #pragma entry#pragma stacksize, #pragma address, #pragma global_register
    There is a possibility that extended specifications will affect program operation.
    Confirm the descriptions on the extended specifications which were output in the message.
  - volatile qualified variables
    The sizes of reads and writes may differ between compilers.
    This compiler may access bit fields with a volatile decorator as a smaller size than the declared type, but the SH compiler will access them as the size of the declared type.
  - Integer promotion of binary operations
    The result of binary operation (such as addition, subtraction, multiplication, division, or comparison) using unsigned int-type and long-type operands may differ from that obtained by using the SH compiler.
    The SH compiler calculates this operation in signed long if the -strict_ansi option is not specified.
    This compiler calculates this operation after converting the operands to the unsigned int type.
  - Types of integer constants exceeding type signed long
    The SH compiler makes values in the range that can be expressed as type unsigned long into type signed long long.
    This compiler makes values in the range that can be expressed as type unsigned long into type unsigned long.
  - Bit field allocation
    The SH compiler does not allocate bits to contiguous areas when the type of a bit field differs from that of the previous bit field.
    This compiler may allocate bits to contiguous areas according to the -Xpack option setting.

- No message will be output for structure and bit field member allocation.
  See "CubeSuite+ Integrated Development Environment User's Manual: RH850 Coding" about declarations that take assignment into account.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Check C program compatibility] in the [C Language] category
  - From the [Individual Compile Options] tab, [Check C program compatibility] in the [C Language] category

[Example of use]

- To check the C source file that has been coded for the SH compiler, describe as:

```
>ccrh -Xcheck=shc -Xcommon=v850e3v5 main.c
```

-Xmisra2004

This option checks source code against the MISRA-C:2004 rules.

[Specification format]

```
-Xmisra2004=string
```

- Interpretation when omitted
  The source code is not checked against the MISRA-C: 2004 rules.

[Detailed description]

- This option checks source code against the MISRA-C:2004 rules.
  A message is output if the item specified for the check is *string*.
- The items that can be specified as *string* are shown below.
  An error will occur if any other item is specified.

Check Item (*string*)	Parameter (*value*)	Description
all	None	The source code is checked against all of the rules which are supported.
apply	*num*[,*num*]...	The source code is checked against the rules with the numbers specified by *num* among the rules which are supported.
ignore	*num*[,*num*]...	The source code is checked against the rules with the numbers that are not specified by *num* among the rules which are supported.
required	None	The source code is checked against the rules of the "required" type among the rules which are supported.
required_add	*num*[,*num*]...	The source code is checked against the rules of the "required" type and the rules with the numbers specified by *num* among the rules which are supported.
required_remove	*num*[,*num*]...	The source code is checked against the rules of the "required" type except for the rules with the numbers specified by *num* among the rules which are supported.
file		The source code is checked against the rules with the numbers described in specified file *file* among the rules which are supported. Specify one rule number per one line in the file.

- The items that can be specified as *num* are shown below.
  An error will occur if any other item is specified.

  2.2 2.3
  4.1 4.2
  5.2 5.3 5.4 5.5 5.6
  6.1 6.2 6.3 6.4 6.5
  7.1
  8.1 8.2 8.3 8.5 8.6 8.7 8.11 8.12
  9.1 9.2 9.3
  10.1 10.2 10.3 10.4 10.5 10.6
  11.1 11.2 11.3 11.4 11.5
  12.1 12.3 12.4 12.5 12.6 12.7 12.8 12.9 12.10 12.11 12.12 12.13

13.1 13.2 13.3 13.4
14.2 14.3 14.4 14.5 14.6 14.7 14.8 14.9 14.10
15.1 15.3 15.4 15.5
16.1 16.3 16.5 16.6 16.9
17.5
18.1 18.4
19.3 19.6 19.7 19.8 19.11 19.13 19.14 19.15
20.4 20.5 20.6 20.7 20.8 20.9 20.10 20.11 20.12

- An error will occur if *string* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Apply rule], [Rule number description file], [Rule number], [Exclusion rule number], [Check rule number besides required rule], and [Exclusion rule number from required rule] in the [MISRA-C:2004 Rule Check] category
  - From the [Individual Compile Options] tab, [Apply rule], [Rule number description file], [Rule number], [Exclusion rule number], [Check rule number besides required rule], and [Exclusion rule number from required rule] in the [MISRA-C:2004 Rule Check] category

[Example of use]

- To check the source code against MISRA-C:2004 rule number: 5.2, 5.3, and 5.4, describe as:

```
>ccrh -Xmisra2004=apply5.2,5.3,5.4 -Xcommon=v850e3v5 main.c
```

-Xignore_files_misra

This option specifies files that will not be checked against the MISRA-C: 2004 rules.

[Specification format]

```
-Xignore_files_misra=file[,file]...
```

- Interpretation when omitted
  All C source files are checked.

[Detailed description]

- This option does not check file *file* against the MISRA-C: 2004 rules.
- This option is valid only when the -Xmisra2004 option is specified.
  If the -Xmisra2004 option is not specified, this option will be ignored (A warning will not be output).
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Rule check exclusion file] in the [MISRA-C:2004 Rule Check] category
  - From the [Individual Compile Options] tab, [Rule check exclusion file] in the [MISRA-C:2004 Rule Check] category

[Example of use]

- Not to check sample.c against the MISRA-C: 2004 rules, describe as:

```
>ccrh -Xmisra2004=all -Xignore_files_misra=sample.c -Xcommon=v850e3v5 main.c
```

-Xcheck_language_extention

This option enables the source-code checking of the MISRA-C:2004 rules, which are partially suppressed by the extended language specifications.

[Specification format]

```
-Xcheck_language_extension
```

- Interpretation when omitted
  The source-code checking of the MISRA-C:2004 rules is disabled, which are partially suppressed by the extended language specifications.

[Detailed description]

- This option enables the source-code checks of the MISRA-C:2004 rules in the following cases where they are suppressed by the unique language specifications extended from the C language standard.
  - When the function has no prototype declaration (rule 8.1) and #pragma interrupt is specified for it.
- This option is valid only when the -Xmisra2004 option is specified.
  If the -Xmisra2004 option is not specified, this option will be ignored (A warning will not be output).
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Output message of the enhanced key word and extended specifications] in the [MISRA-C:2004 Rule Check] category
  - From the [Individual Compile Options] tab, [Output message of the enhanced key word and extended specifications] in the [MISRA-C:2004 Rule Check] category

[Example of use]

- To enable the source-code checking of the MISRA-C:2004 rules, which are partially suppressed by the extended language specifications, describe as:

```
>ccrh -Xmisra2004=all -Xcheck_language_extension -Xcommon=v850e3v5 main.c
```

Japanese/Chinese character control

The Japanese/Chinese character control option is as follows.

- -Xcharacter_set

-Xcharacter_set

This option specifies the Japanese/Chinese character code.

[Specification format]

```
-Xcharacter_set=code
```

- Interpretation when omitted
  Processing of Japanese/Chinese character encoding is not performed.

[Detailed description]

- This option specifies the character code to be used for Japanese/Chinese comments and character strings in the source file.
- The items that can be specified as *code* are shown below.
  An error will occur if any other item is specified.
  Operation is not guaranteed if the specified character code differs from the character code of the source file.

none	Does not process the Japanese and Chinese character code
euc_jp	EUC (Japanese)
sjis	SJIS
utf8	UTF-8
big5	Traditional Chinese
gb2312	Simplified Chinese

- An error will occur if *code* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Character encoding] in the [Character Encoding] category
  - From the [Individual Compile Options] tab, [Character encoding] in the [Character Encoding] category

[Example of use]

- To specify EUC as the character code to be used for Japanese comments and character strings in the input file, describe as:

```
>ccrh -Xcharacter_set=euc_jp -Xcommon=v850e3v5 main.c
```

Optimization specification

The optimization specification options are as follows.

- -O
- -Xintermodule
- -Xinline_strcpy
- -Xmerge_string
- -Xalias
- -Xmerge_files
- -Xwhole_program

-O

This option specifies the optimization level or the details of each optimization items.

[Specification format]

```
-O[level]
-O[item[=value][,item[=value]]...]
```

- Interpretation when omitted
  Only optimization that debugging is not affected is performed (It is the same result as when -Odefault option is specified).

[Detailed description]

- This option specifies the optimization level or the details of each optimization items.
- The items that can be specified as *level* are shown below.
  An error will occur if any other item is specified.

nothing	Optimization with debugging precedence Regards debugging as important and suppresses all optimization including default optimization.
default	Default Performs optimization that debugging is not affected (optimization of expressions and register allocation, and the like).
size	Optimization with the object size precedence Regards reducing the ROM/RAM capacity as important and performs the maximum optimization that is effective for general programs.
speed	Optimization with the execution speed precedence Regards shortening the execution speed as important and performs the maximum optimization that is effective for general programs.

- If *level* is omitted, it is assumed that "size" has been specified.
- The items that can be specified as *item* and *value* are shown below.
  An error will occur if any other item is specified.

Optimization Item (*item*)	Parameter (*value*)	Description
unroll	0 to 4294967295 (Integer value)	Loop expansion When 0 or 1 is specified, the expansion is suppressed. If 2 or more is specified, the number of executions is converted into *N* loops (*N* is a constant), and a loop including code expanded *value* times. If the code size after expansion is too great or if the number of times of execution of the loop is too few, the number of times of expansion may decrease, or the loop may not be expanded at all. In addition, a loop having a complicated structure, such as having inner loops, may not be expanded. If *value* is omitted, it is assumed that 4 has been specified. If the -Ospeed option is specified, this item is assumed that the -Ounroll=4 option is specified.
inline	0 to 3 (Integer value)	Inline expansion for functions *value* signifies the level of the expansion. 0: Suppresses all inline expansion including the function for which "#pragma inline" is specified. 1: Performs inline expansion for only a function for which "#pragma inline" is specified. 2: Distinguishes a function that is the target of expansion automatically and expands it. 3: Distinguishes the function that is the target of expansion automatically and expands it, while minimizing the increase in code size. However, if 1 to 3 is specified, the function that is specified by "#pragma inline" may not be expanded according to the content of the function and the status of compilation. If *value* is omitted, it is assumed that 2 has been specified. This item is valid when the -Osize or -Ospeed option is specified (when the -Osize option is specified, it is assumed that the -Oinline=3 option has been specified. When the -Ospeed option is specified, it is assumed that the -Oinline=2 option has been specified). If any of the -Osize, -Ospeed or -Oinline option is not specified, this item is assumed that the -Oinline=1 option is specified. If the -Onothing option is specified, this item is assumed that the -Oinline=0 option is specified.
inline_size	0 to 65535 (Integer value)	Size for inline expansion Specify the maximum increasing rate (%) of the code size up to which inline expansion is performed. If *value* is omitted, it is assumed that 100 has been specified. This item is valid when the -Oinline=2 option is specified (including when the -Ospped option is specified).
delete_static_func	on or off	Deleting unused static functions If *value* is omitted, it is assumed that "on" has been specified. This item is valid when the -Onothing option is not specified.
pipeline	on or off	Pipeline optimization If *value* is omitted, it is assumed that "on" has been specified. This item is always valid when the -Ospeed option is specified.

Optimization Item (*item*)	Parameter (*value*)	Description
tail_call	on or off	Converting function calls at the end of functions to jr If "on" is specified, then if there is a function call at the end of a function, and certain conditions are met, a jr instruction will be generated for that call rather than a jarl instruction. The lp store/restore code will be removed, reducing the code size. However, some debug functions cannot be used. If *value* is omitted, it is assumed that "on" has been specified. This item is valid when the -Ospeed or -Osize option is specified.
map	*file name*	External variable access optimization Base addresses are set in accordance with the external symbol allocation information generated by the linker, and code for accessing external and static variables relative to the base addresses is generated. When symbol "__gp_data" is defined and a value is specified in gp in the code of the startup routine, code for accessing variables relative to gp when possible is generated in accordance with the external symbol allocation information. Specify the external symbol allocation information file generated by the optimizing linker as *file name*. If *file name* is omitted, it performs linking once, and then after creating the external symbol allocation information file, repeats the process from compilation to linking.
smap	None	Optimization of access to external variables defined in the compilation unit Base address is set for external and static variables defined in the file to be compiled, and code that accesses these relative to the base address is generated.

- If *item* is omitted, it is assumed that this option has specified "-Osize".
- If this option is specified more than once for the same *item*, the option specified last will be valid.
- When -O*item* is not specified, the optimization items are interpreted as follows according to the -O*level* setting.

Optimization Item (*item*)	Optimization Level (*level*)			
	nothing	default	size	speed
unroll	1	1	1	4
inline	0	1	3	2
inline_size	-	-	-	100
delete_static_func	off	on	on	on
pipeline	off	off	off	on
tail_call	off	off	on	on
map	-	-	-	-
smap	-	-	-	-

- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Level of optimization] in the [Frequently Used Options(for Compile)] category
  - From the [Compile Options] tab, [Level of optimization] in the [Optimization] category, [Maximum number of loop expansions], [Remove unused static functions], [Perform inline expansion], [Maximum increasing rate of

inline expansion size], [Perform pipeline optimization], [Use jr instruction to call a function at the end of the function], and [Optimize accesses to external variables] in the [Optimization(Details)] category

- From the [Individual Compile Options] tab, [Level of optimization] in the [Optimization] category, [Maximum number of loop expansions], [Remove unused static functions], [Perform inline expansion], [Maximum increasing rate of inline expansion size], [Perform pipeline optimization], [Use jr instruction to call a function at the end of the function], and [Optimize accesses to external variables] in the [Optimization(Details)] category

[Example of use]

- To perform optimization with the object size precedence, describe as:

```
>ccrh -Osize -Xcommon=v850e3v5 main.c
```

-Xintermodule

This option performs inter-module optimization.

[Specification format]

```
-Xintermodule
```

- Interpretation when omitted
  Inter-module optimization is not performed.

[Detailed description]

- This option performs inter-module optimization.
- The main optimization contents are shown below.
  - Optimization using inter-procedural alias analysis
    The example of the output code is shown below.

```
[C source]
extern int x[2];
static int func1(int *a, int *b) {
    *a=0;
    *b=1;
    return *a;
}
int func2() {
    return func1(&x[0], &x[1]);
}

[Output assembler source]
_func1.1:
    .stack  _func1.1 = 0
    mov     #_x, r2
    st.w    r0, 0x00000000[r2]
    mov     0x00000001, r5
    st.w    r5, 0x00000004[r2]
    mov     0x00000000, r10     ; "0" is assigned directly because "a" and "b" point to
different types.
    jmp     [r31]
_func2:
    .stack  _func2 = 0
    mov     #_x, r6
    addi    0x00000004, r6, r7
    br9     _func1.1
```

- Constant propagation of parameters and return values
  The example of the output code is shown below.

```
[C source]
static int func(int x, int y, int z) {
    return x-y+z;
}
int func2() {
    return func(3,4,5);
}

[Output assembler source]
_func.1:
    .stack  _func.1 = 0
    mov     0x00000000, r10     ; "4(=3-4+5)" is assigned directly.
    jmp     [r31]
_func2:
    .stack  _func2 = 0
    mov     0x00000005, r8
    mov     0x00000004, r7
    mov     0x00000003, r6
    br9     _func.1
```

- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Perform inter-module optimization] in the [Optimization(Details)] category
  - From the [Individual Compile Options] tab, [Perform inter-module optimization] in the [Optimization(Details)] category

[Example of use]

- To perform inter-module optimization on source files "main.c" and "sub.c", describe as:

```
>ccrh -Xintermodule -Xwhole_program -Xcommon=v850e3v5 main.c sub.c
```

-Xinline_strcpy

This option performs inline expansion of standard library functions "strcpy", "strcmp", "memcpy", and "memset" calls.

[Specification format]

```
-Xinline_strcpy
```

- Interpretation when omitted
  Inline expansion of standard library functions "strcpy", "strcmp", "memcpy", and "memset" calls is not performed.

[Detailed description]

- This option performs inline expansion of standard library functions "strcpy", "strcmp", "memcpy", and "memset" calls.
- This option can not be specified together with the -Xpack option.
- Inline expansion of strcpy is performed only when the second argument is a character string.
- If this option is specified, arrays and character strings are allocated automatically to 4-byte boundary area.
- This improves the execution speed of the program to be generated, but it increases the code size.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Perform inline expansion of strcpy/strcmp/memcpy/memset] in the [Optimization(Details)] category
  - From the [Individual Compile Options] tab, [Perform inline expansion of strcpy/strcmp/memcpy/memset] in the [Optimization(Details)] category

[Example of use]

- To perform inline expansion of standard library functions "strcpy", "strcmp", "memcpy", and "memset" calls, describe as:

```
>ccrh -Xinline_strcpy -Xcommon=v850e3v5 main.c
```

-Xmerge_string

This option merges string literals.

[Specification format]

```
-Xmerge_string
```

- Interpretation when omitted
  If the same string literals are included multiple times in the source file, each will be allocated to a separate area.

[Detailed description]

- When the same string literals exist in the source file, this option merges them and allocates to the one area.
- The same string literals are allocated to the same area, regardless of whether #pragma section sconst (const) is specified.
  However, if a different section is specified, the section to which the string literal is allocated will depend on the order of appearance in the source.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Merge string literals] in the [Optimization(Details)] category
  - From the [Individual Compile Options] tab, [Merge string literals] in the [Optimization(Details)] category

[Example of use]

- When the same string literals exist in the source file, to merge them and allocate to the one area, describe as:

```
>ccrh -Xmerge_string -Xcommon=v850e3v5 main.c
```

-Xalias

This option performs optimization with consideration for the type of the data indicated by the pointer.

[Specification format]

```
-Xalias=value
```

- Interpretation when omitted
  Optimization with consideration for the type of the data indicated by the pointer, based on the ANSI standard is not performed.

[Detailed description]

- This option specifies whether to perform optimization with consideration for the type of the data indicated by the pointer, based on the ANSI standard.
- The items that can be specified as *value* are shown below.
  An error will occur if any other item is specified.

ansi	Optimization with consideration for the type of the data indicated by the pointer, based on the ANSI standard is performed.
noansi	Optimization with consideration for the type of the data indicated by the pointer, based on the ANSI standard is not performed.

- An error will occur if *value* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Perform optimization considering type of data indicated by pointer] in the [Optimization(Details)] category
  - From the [Individual Compile Options] tab, [Perform optimization considering type of data indicated by pointer] in the [Optimization(Details)] category

[Example of use]

- To perform optimization with consideration for the type of the data indicated by the pointer, based on the ANSI standard, describe as:

```
>ccrh -Xalias=ansi -Xcommon=v850e3v5 main.c
```

-Xmerge_files

This option merges two or more C source files and compiles them.

[Specification format]

```
-Xmerge_files
```

- Interpretation when omitted
  Compilation is performed at the input-file level, without merging.

[Detailed description]

- This option merges two or more C source files and compiles them. And then it outputs one file.
- If the -o option is specified, then the specified file name is used for the output file. If the -o option is not specified, then the file name is in accordance with the interpretation of the -o option being omitted for the initially specified C source file.
- If one C source file is input and if this option is specified together with the -P option, this option will be invalid.
- If this option is specified at the same time as the -S or -c option, then for the 2nd and subsequent C source files that are specified, an empty file is output in accordance with the interpretation that the -o option was omitted.
- If this option is specified at the same time with the -Oinline option, inline expansion is performed between files.
- Operation is not guaranteed if an object file is generated with this option specified and any of link options -delete, -rename, and -replace is specified at linkage of the object file.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Perform inter-module optimization] in the [Optimization(Details)] category
  - From the [Individual Compile Options] tab, [Perform inter-module optimization] in the [Optimization(Details)] category

[Example of use]

- To merge main.c and sub.c, compile them, and output one file, describe as:

```
>ccrh -Xmerge_files -Xwhole_program -Xcommon=v850e3v5 main.c sub.c
```

-Xwhole_program

This option performs optimization assuming that the files to be compiled comprise the entire program.

[Specification format]

```
-Xwhole_program
```

- Interpretation when omitted
  It is not assumed that the files to be compiled comprise the entire program.

[Detailed description]

- This option performs optimization assuming that the files to be compiled comprise the entire program.
- The compilation is performed assuming that the following conditions are met. Operation is not guaranteed if these conditions are not met.
  - The values and addresses of extern variables defined in the files to be compiled will not be modified or referenced from outside those files.
  - Even if a file to be compiled calls a function defined outside the files to be compiled, the called function will never call a function in the files to be compiled.
- If this option is specified, it is assumed that the -Xintermodule option is specified.
  If two or more C source files are input, it is assumed that the -Xmerge_files option is specified.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Perform inter-module optimization] in the [Optimization(Details)] category
  - From the [Individual Compile Options] tab, [Perform inter-module optimization] in the [Optimization(Details)] category

[Example of use]

- To perform optimization assuming that the files to be compiled comprise the entire program, describe as:

```
>ccrh -Xwhole_program -Xcommon=v850e3v5 main.c
```

Generated code control

The generated code control options are as follows.

- -Xpack
- -Xbit_order
- -Xpass_source
- -Xswitch
- -Xreg_mode
- -Xreserve_r2
- -Xep
- -Xfloat
- -Xcall_jump
- -Xfar_jump
- -Xdiv
- -Xcheck_div_ov
- -Xuse_fmaf
- -Xunordered_cmpf
- -Xmulti_level
- -Xpatch

-Xpack

This option performs the structure packing.

[Specification format]

```
-Xpack=num
```

- Interpretation when omitted
  The structure packing is not performed.

[Detailed description]

- This option performs the structure packing.
- If this option is specified, struct members will not be aligned by their member types, but rather code will be generated with alignment packed to the specified *num* bytes.
- 1, 2, or 4 can be specified as *num*.
  An error will occur if any other item is specified.
- This option can not be specified together with the -Xinline_strcpy option.
- If this option is specified when the structure packing is specified by the #pragma directive in the C source, the value specified by this option is applied to all structures until the first #pragma directive appears.
  After that, the value of the #pragma directive is applied.
  Even after the #pragma directive has appeared, however, the value specified by the option is applied if the default value is specified (if the value of the packing by the #pragma directive).
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Structure packing] in the [Output Code] category

[Example of use]

- To generate code with struct member alignment packed to 1 byte, describe as:

```
>ccrh -Xpack=1 -Xcommon=v850e3v5 main.c
```

-Xbit_order

This option specifies the order of bit-field members.

[Specification format]

```
-Xbit_order=pos
```

- Interpretation when omitted
  The bit-field members are allocated from the lower bit.

[Detailed description]

- This option specifies the order of bit-field members.
- The items that can be specified as *pos* are shown below.
  An error will occur if any other item is specified.

left	The members are allocated from the upper bit.
right	The members are allocated from the lower bit.

- An error will occur if *pos* is omitted.
- If this option is specified when the order of bit-field members is specified by the #pragma directive in the C source, the value specified by this option is applied to all members until the first #pragma bit_order directive appears. After that, the value of the #pragma directive is applied.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Order of bit-field members] in the [Output Code] category

[Example of use]

- To allocate the bit-field members from the upper bit, describe as:

```
>ccrh -Xbit_order=left -Xcommon=v850e3v5 main.c
```

-Xpass_source

This option outputs a C source program as a comment to the assembly source file.

[Specification format]

```
-Xpass_source
```

- Interpretation when omitted
  The C source program is not output as a comment to the assembly source file.

[Detailed description]

- This option outputs a C source program as a comment to the assembly source file.
- The output comments are for reference only and may not correspond exactly to the code.
  Additionally, non-executed lines may not be output as comments (e.g. type declarations and labels).
  For example, comments concerning global variables, local variables, function declarations, etc., may be output to incorrect positions.
  By specifying the optimization options, the code may be deleted and only the comment may remain.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Output comment to assembly source file] in the [Output Code] category
  - From the [Individual Compile Options] tab, [Output comment to assembly source file] in the [Output Code] category

[Example of use]

- To output a C source program as a comment to the assembly source file, describe as:

```
>ccrh -Xpass_source -S -Xcommon=v850e3v5 main.c
```

-Xswitch

This option specifies the format in which the code of switch statements is to be output.

[Specification format]

```
-Xswitch=type
```

- Interpretation when omitted
  ccrh selects the optimum output format for each switch statement.

[Detailed description]

- This option specifies the format in which the code of switch statements is to be output.
- The items that can be specified as *type* are shown below.
  An error will occur if any other item is specified.

ifelse	Outputs the code in the same format as the if-else statement along a string of case statements. Select this item if the case statements are written in the order of frequency or if only a few labels are used. Because the case statements are compared starting from the top, unnecessary comparison can be reduced and the execution speed can be increased if the case statement that most often matches is written first.
binary	Outputs the code in the binary search format. Searches for a matching case statement by using a binary search algorithm. If this item is selected when many labels are used, any case statement can be found at almost the same speed.
table	Outputs the code in a table jump format. References a table indexed on the values in the case statements, and selects and processes case labels from the switch statement values. The code will branch to all the case statements with about the same speed. However, if case values are not used in succession, an unnecessary area will be created. If the difference between the maximum and minimum values of the case labels exceeds 8192, then this option is ignored, and the optimum output format for each switch statement is selected automatically.

- An error will occur if *type* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Output code of switch statement] in the [Output Code] category
  - From the [Individual Compile Options] tab, [Output code of switch statement] in the [Output Code] category

[Example of use]

- To output a code for the switch statement in the binary search format, describe as:

```
>ccrh -Xswitch=binary -Xcommon=v850e3v5 main.c
```

-Xreg_mode

This option specifies the register mode.

[Specification format]

```
-Xreg_mode=mode
```

- Interpretation when omitted
  The 32-register mode object file is generated.

[Detailed description]

- This option generates the object file for the specified register mode.
- This option limits the number of registers used by ccrh to 32 (the 32-register mode) or 22 (the 22-register mode or register mode "common") and embeds the magic number into the object file.
- Use register mode "common" to generate the object file that does not depend on register modes.
- The items that can be specified as *mode* are shown below.
  An error will occur if any other item is specified.

Register Mode (*mode*)	Working Registers	Registers for Register Variables
common	r10 to r14	r25 to r29
22	r10 to r14	r25 to r29
32	r10 to r19	r20 to r29

- An error will occur if *mode* is omitted.
- This option generates the code using the register that can be used for a C source file.
- An abort error will occur if the register that cannot be used for the assembly source file.
- The same register mode must be specified for all source files.
  The different register mode cannot be specified for each source file.
  If there are object files with different register modes, an error will occur during linking.
- By specifying this option, the register mode of the software register bank function can be changed.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Register mode] in the [Register Mode] category

[Example of use]

- To generate the 22-register mode object file, describe as:

```
>ccrh -Xreg_mode=22 -Xcommon=v850e3v5 main.c
```

-Xreserve_r2

This option reserves the r2 register.

[Specification format]

```
-Xreserve_r2
```

- Interpretation when omitted
  The compiler uses the r2 register without reserving it.

[Detailed description]

- This option reserves the r2 register and generates code that does not use this register by the compiler.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Reserve r2 register] in the [Register Mode] category

[Example of use]

- To reserve the r2 register and generates code that does not use this register by the compiler, describe as:

```
>ccrh -Xreserve_r2 -Xcommon=v850e3v5 main.c
```

-Xep

This option specifies how to handle the ep register.

[Specification format]

```
-Xep=mode
```

- Interpretation when omitted
  The ep register is treated as a register guaranteeing the value before and after the function call.

[Detailed description]

- This option specifies how to handle the ep register.
- The items that can be specified as *mode* are shown below.
  An error will occur if any other item is specified.

fix	Fixes the value of the ep register for the entire project. Specify this parameter when any of the following attribute strings in a #pragma section in the project is used. ep_auto, ep_disp4, ep_disp5, ep_disp7, ep_disp8, ep_disp16, ep_disp23
callee	Treats the ep register as a register guaranteeing the value before and after the function call. Specify this parameter when the -Omap or -Osmap option is specified.

- An error will occur if *mode* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [ep-register treatment] in the [Register Mode] category

[Example of use]

- To fix the value of the ep register for the entire project, describe as:

```
>ccrh -Xep=fix -Xcommon=v850e3v5 main.c
```

-Xfloat

This option controls generating floating-point calculation instructions.

[Specification format]

```
-Xfloat=type
```

- Interpretation when omitted
  If the -Xcpu=g3m option is specified and the -Xcpu option is not specified, floating-point calculation instructions are generated.
  If the -Xcpu=g3k option is specified, runtime function call instructions are generated.

[Detailed description]

- This option controls generating floating-point calculation instructions.
- The items that can be specified as *type* are shown below.
  An error will occur if any other item is specified.

soft	Generates runtime library call instructions for floating-point calculations. Additionally, the FPU math support library will not be linked.
fpu	Generates floating-point calculation instructions of FPU (floating-point unit) for floating-point calculations. Additionally, the FPU math support library will be linked.

- An error will occur if *type* is omitted.
- If this option is specified together with the -Xcpu=g3k option, this option will be invalid and runtime function call instructions will always be generated.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Floating-point calculating type] in the [Output Code] category
  - From the [Individual Compile Options] tab, [Floating-point calculating type] in the [Output Code] category

[Example of use]

- To generate the jarl32 and jr32 instructions for function-call branches, describe as:

```
>ccrh -Xfloat=soft -Xcommon=v850e3v5 main.c
```

-Xcall_jump

This option controls generating function-call branch instructions.

[Specification format]

```
-Xcall_jump=num
```

- Interpretation when omitted
  The jarl and jr instructions are generated for function-call branches.

[Detailed description]

- This option controls generating function-call branch instructions.
- The items that can be specified as *num* are shown below.
  An error will occur if any other item is specified.

22	Generates the jarl and jr instructions for the branch to the function.
32	Generates the jarl32 and jr32 instructions for the branch to the function.

- An error will occur if *num* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Specify jump instruction] in the [Output Code] category
  - From the [Individual Compile Options] tab, [Specify jump instruction] in the [Output Code] category

[Example of use]

- To generate the jarl32 and jr32 instructions for the branch to the function, describe as:

```
>ccrh -Xcall_jump=32 -Xcommon=v850e3v5 main.c
```

-Xfar_jump

This option controls outputting far jump.

[Specification format]

```
-Xfar_jump=file
```

- Interpretation when omitted
  The instructions in accordance with the -Xcall_jump option are generated.

[Detailed description]

- This option generates the code that uses the jarl32 and jr32 instruction for the branch to functions (including interrupt functions described by C language) specified in far jump calling function list file *file*.
- ".fjp" is recommended as the extension of *file*.
- An error will occur if *file* does not exist.
- An error will occur if *file* is omitted.
- An error will occur during linking if the function is in the range that cannot be branched to by the jarl or jr directive (±2MB or more) when the -Xcall_jump=22 option is specified. In this case, recompile by using this option.
- If this option is specified more than once, the option specified last will be valid.
- The example of the output code is shown below.
  - C source

```
far_func();     /* "jarl _far_func, lp" is output by default. */
```

  - Output assembly source

```
jarl32  _far_func, lp
```

- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Far Jump file names] in the [Output Code] category
  - From the [Individual Compile Options] tab, [Far Jump file names] in the [Output Code] category

Remark Cautions about are the format of the far jump calling function list file as follows.
- Describe with one function name per line.
  If two or more function name is described, the first name will be valid.
- Describe the function name (label name in an assembly source) by prefixing "_" to that in C language.
  However, the following formats can be specified instead of function names.

Format	Meaning
{all function}	All functions are called.
{all interrupt}	All interrupt functions are called.

- Rutime routines can be specified instead of functions.
  In this case, describe the runtime routine name as-is as the function name.
- A space and tab can be inserted before and after function names.

- Only ASCII characters can be used.
  After the space characters at the beginning of a line, a non-space character string until the next space character or the end of the line is treated as a function name, and the rest of the line (from the space character to the end of the line) is ignored.
- Comments cannot be inserted.
- Up to 1023 characters can be specified per line (including a space and tab).

The example when specifying functions is shown below.

```
_func_led
_func_beep
_func_motor
    :
_func_switch
```

[Example of use]

- To generate the code that uses jmp directive to branch to the function specified in func.fjp, describe as:

```
>ccrh -Xfar_jump=func.fjp -Xcommon=v850e3v5 main.c
```

-Xdiv

This option generates the div and divu instructions for division.

[Specification format]

```
-Xdiv
```

- Interpretation when omitted
  The divq and divqu instructions are generated for division.

[Detailed description]

- This option generates the div and divu instructions instead of the divq and divqu instructions for division.
- Although the divq and divqu instructions are fast, the number of execution cycles will differ depending on the values of the operands.
  For this reason, specify this option if it is necessary to maintain a constant number of execution cycles at all times (e.g. in order to guarantee real-time performance).
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Generate div/divu instructions] in the [Output Code] category
  - From the [Individual Compile Options] tab, [Generate div/divu instructions] in the [Output Code] category

[Example of use]

- To generate the div and divu instructions for division, describe as:

```
>ccrh -Xdiv -Xcommon=v850e3v5 main.c
```

-Xcheck_div_ov

This option checks the OV flag at division.

[Specification format]

```
-Xcheck_div_ov=num
```

- Interpretation when omitted
  Code that does not check the OV flag at division is generated.

[Detailed description]

- This option generates code (fetrap instruction) that checks the OV flag after division instructions and generate an FE level software exception when the OV flag is 1.
- The value that can be specified for *num* is 1 to 15 (a value that can be specified for the operand of the fetrap instruction).
  An error will occur if any other item is specified.
- An error will occur if *num* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Generate OV flag check code in division operation], [Vector number of fetrap instruction in divide exception] in the [Output Code] category
  - From the [Individual Compile Options] tab, [Generate OV flag check code in division operation], [Vector number of fetrap instruction in divide exception] in the [Output Code] category

[Example of use]

- To check the OV flag at division, describe as:

```
>ccrh -Xcheck_div_ov=1 -Xcommon=v850e3v5 main.c
```

-Xuse_fmaf

This option generates product-sum operation instructions.

[Specification format]

```
-Xuse_fmaf
```

- Interpretation when omitted
  Product-sum operation instructions are not generated.

[Detailed description]

- This option generates product-sum operation instructions (fmaf.s, fmsf.s, fnmaf.s, and fnmsf.s) for single-precision floating-point product-sum operations.
- Specifying this option will accelerate the execution speed but change the operation precision.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Generate product-sum operation instruction] in the [Output Code] category
  - From the [Individual Compile Options] tab, [Generate product-sum operation instruction] in the [Output Code] category

[Example of use]

- To generate product-sum operation instructions for single-precision floating-point product-sum operations, describe as:

```
>ccrh -Xuse_fmaf -Xcommon=v850e3v5 main.c
```

-Xunordered_cmpf

This option detects invalid operation exceptions in floating-point comparison.

[Specification format]

```
-Xunordered_cmpf
```

- Interpretation when omitted
  Invalid operation exceptions in floating-point comparison are not detected.

[Detailed description]

- This option generates code by using the comparison condition for generating an invalid operation exception when any of the comparison values is a not-a-number in floating-point comparison.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Check invalid exception in cmpf instruction] in the [Output Code] category
  - From the [Individual Compile Options] tab, [Check invalid exception in cmpf instruction] in the [Output Code] category

[Example of use]

- To detect invalid operation exceptions in floating-point comparison, describe as:

```
>ccrh -Xunordered_cmpf -Xcommon=v850e3v5 main.c
```

-Xmulti_level

This option specifies the generation of a multi-core program.

[Specification format]

```
-Xmulti_level=level
```

- Interpretation when omitted
  A single-core program is generated.

[Detailed description]

- This option generates a program for the specified core.
- The items that can be specified as *level* are shown below.
  An error will occur if any other item is specified.

0	Generates a single-core program. The #pragma pmodule directives in the program are ignored.
1	Generates a multi-core program. The #pragma pmodule directives in the program become valid.

- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Type of a generating program] in the [Output Code] category
  - From the [Individual Compile Options] tab, [Type of a generating program] in the [Output Code] category

[Example of use]

- To generate a multi-core program, describe as:

```
>ccrh -Xmulti_level=1 -Xcommon=v850e3v5 file1.c file2.c
```

-Xpatch

This option applies a patch.

[Specification format]

```
-Xpatch=string
```

- Interpretation when omitted
  The patch is not applied.

[Detailed description]

- "dw_access" can be specified as *string*.
  An error will occur if any other item is specified.
- When the -Xpatch=dw_access option is specified, a code is generated without using the ld.dw and st.dw instructions.

[Example of use]

- To generate a code without using the ld.dw and st.dw instructions, describe as:

```
>ccrh -Xpatch=dw_access -Xcommon=v850e3v5 main.c
```

Information file output control

The information file output control option is as follows.

- -Xcref

-Xcref

This option outputs the static analysis information file.

[Specification format]

```
-Xcref=path
```

- Interpretation when omitted
  The static analysis information file is not output.

[Detailed description]

- This option specifies the location where the static analysis information file to be generated during compilation as *path*.
- If an existing folder is specified as *path*, the static analysis information file is saved under the C source file name with the extension replaced by ".cref" to *path*.
- If an existing file name is specified or a non-existing folder or file name is specified, the static analysis information file is output with *path* as the file name when one static analysis information file is output.
  If two or more static analysis information files are output, an error will occur.
- An error will occur if "=*path*" is omitted.
- If two or more files with the same name (even if they are in different folders) are specified as source files, then a warning is output, and a static analysis information file is only saved for the last source file to be specified.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Output cross reference information] in the [Output File Type and Path] category

[Example of use]

- To output the static analysis information file as file name "info.cref", describe as:

```
>ccrh -Xcref=info.cref -Xcommon=v850e3v5 main.cs
```

Error output control

The error output control options are as follows.

- -Xerror_file

-Xerror_file

This option outputs error messages to a file.

[Specification format]

```
-Xerror_file=file
```

- Interpretation when omitted
  Error messages are output to only the standard error output.

[Detailed description]

- This option outputs error messages to the standard error output and file *file*.
- If *file* already exists, it will be overwritten.
- An error will occur if *file* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Output error message file], [Error message file output folder], [Error message file name] in the [Error Output] category
  - From the [Individual Compile Options] tab, [Output error message file], [Error message file output folder], [Error message file name] in the [Error Output] category

[Example of use]

- To output error messages to the standard error output and file "err", describe as:

```
>ccrh -Xerror_file=err -Xcommon=v850e3v5 main.c
```

Warning message output control

The warning message output control options are as follows.

- -Xno_warning

-Xno_warning

This option suppresses outputting warning messages of the specified number.

[Specification format]

```
-Xno_warning=num[,num]...
-Xno_warning=num1-num2
```

- Interpretation when omitted
  All warning messages are output.

[Detailed description]

- This option suppresses outputting warning messages of the specified number.
- Specify the error numbers as *num*, *num1*, and *num2*.
  If the error number that does not exist, it will be ignored.
- An error will occur if *num*, *num1*, or *num2* is omitted.
- If *num1-num2* is specified, it is assumed that error numbers within the range have been specified.
- The error number specified by this option is the rightmost 5 digits of the 7-digit number following the "W".
  See "CubeSuite+ Integrated Development Environment User's Manual: Message" for error numbers.
- This option controls the warning messages of the compiler and assembler and does not control the messages of the optimizing linker.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Undisplayed warning message] in the [Warning Message] category
  - From the [Individual Compile Options] tab, [Undisplayed warning message] in the [Warning Message] category

[Example of use]

- To suppress outputting warning message "W0511146" and "W0511147", describe as:

```
>ccrh -Xno_warning=11146,11147 -Xcommon=v850e3v5 main.c
```

Phase individual option specification

The phase individual option specification options are as follows.

- -Xasm_option
- -Xlk_option

-Xasm_option

This option specifies assemble options.

[Specification format]

```
-Xasm_option=arg
```

- Interpretation when omitted
  The ccrh driver interprets all specified options.

[Detailed description]

- This option passes *arg* to the assembler as the assemble option.
- Use this option to pass a file to the assembler containing an identifier that the ccrh driver does not recognize as an assembly source file.
- An error will occur if *arg* is a non-existent assemble option.
- An error will occur if *arg* is omitted.

[Example of use]

- To pass the -Xasm_far_jump option to the assembler, describe as:

```
>ccrh -Xasm_option=-Xasm_far_jump -Xcommon=v850e3v5 main.c
```

The example above has the same meaning as the following.

```
>ccrh -S -Xcommon=v850e3v5 main.c
>asrh -Xasm_far_jump -Xcommon=v850e3v5 main.asm
```

-Xlk_option

This option specifies link options.

[Specification format]

```
-Xlk_option=arg
```

- Interpretation when omitted
  The ccrh driver interprets all specified options.

[Detailed description]

- This option passes *arg* to the linker as the link option.
- Use this option to pass a file to the linker containing an identifier that the ccrh driver does not recognize as input to the linker.
- An error will occur if *arg* is a non-existent link option.
- An error will occur if *arg* is omitted.

[Example of use]

- To pass the -form=relocate option to the linker, describe as:

```
>ccrh -Xlk_option=-form=relocate -Xcommon=v850e3v5 main.c
```

The example above has the same meaning as the following.

```
>ccrh -c -Xcommon=v850e3v5 main.c
>rlink -form=relocate main.obj
```

Subcommand file specification

The subcommand file specification option is as follows.

- @

@

This option specifies a subcommand file.

[Specification format]

```
@file
```

- Interpretation when omitted
  Only the options and file names specified on the command line are recognized.

[Detailed description]

- This option handles *file* as a subcommand file.
- An error will occur if *file* does not exist.
- An error will occur if *file* is omitted.
- See "B.4.2 Subcommand file usage" for details about a subcommand file.

[Example of use]

- To handle "command.txt" as a subcommand file, describe as:

```
>ccrh @command.txt -Xcommon=v850e3v5
```

B.5.2 Assemble options

This section explains options for the assemble phase.

Caution about options are shown below.

- Uppercase characters and lowercase characters are distinguished for options.
- When numerical values are specified as parameters, decimal or hexadecimal numbers which starts with "0x" ("0X") can be specified.
  Uppercase characters and lowercase characters are not distinguished for the alphabet of hexadecimal numbers.
- When a file name is specified as a parameter, it can include the path (absolute path or relative path).
  When a file name without the path or a relative path is specified, the reference point of the path is the current folder.
- When a parameter includes a space (such as a path name), enclose the parameter in a pair of double quotation marks (").
- When the -Xprn_path or -Xasm_far_jump option is specified for ccrh command, the -Xasm_option option must be used.

The types and explanations for options are shown below.

Table B-3. Assemble Options

Classification	Option	Description
Version/help display specification	-V	This option displays the version information of asrh.
	-h	This option displays the descriptions of asrh options.
Output file specification	-o	This option specifies the output file name.
	-Xobj_path	This option specifies the folder to save an object file generated during assembling.
	-Xprn_path	This option specifies the folder to save the assemble list file.
Source debugging control	-g	This option outputs information for source debugging.
Device specification	-Xcommon	This option specifies that an object file common to the various devices is generated.
	-Xcpu	This option specifies that an object for the specified core is generated.
Preprocessor control	-D	This option defines assembler symbols.
	-U	This option deletes the assembler symbol definition by the -D option.
	-I	This option specifies the folder to search include files.
Japanese/Chinese character control	-Xcharacter_set	This option specifies the Japanese/Chinese character code.
Generated code control	-Xreg_mode	This option specifies the register mode.
	-Xreserve_r2	This option reserves the r2 register.
	-Xep	This option specifies how to handle the ep register.
Assembler control specification	-Xasm_far_jump	This option controls outputting far jump for an assembly source file.
Error output control	-Xerror_file	This option outputs error messages to a file.
Warning message output control	-Xno_warning	This option suppresses outputting warning messages of the specified number.

Classification	Option	Description
Subcommand file specification	@	This option specifies a subcommand file.

Version/help display specification

The version/help display specification options are as follows.

- -V
- -h

-V

This option displays the version information of asrh.

[Specification format]

```
-V
```

- Interpretation when omitted
  Assembling is performed without displaying the version information of asrh.

[Detailed description]

- This option outputs the version information of asrh to the standard error output.
  It does not execute assembling.

[Example of use]

- To output the version information of asrh to the standard error output, describe as:

```
>asrh -V -Xcommon=v850e3v5
```

-h

This option displays the descriptions of asrh options.

[Specification format]

```
-h
```

- Interpretation when omitted
  The descriptions of asrh options are not displayed.

[Detailed description]

- This option outputs the descriptions of asrh options to the standard error output.
  It does not execute assembling.

[Example of use]

- To output the descriptions of asrh options to the standard error output, describe as:

```
>asrh -h -Xcommon=v850e3v5
```

Output file specification

The output file specification options are as follows.

- -o
- -Xobj_path
- -Xprn_path

-o

This option specifies the output file name.

[Specification format]

```
-ofile
```

- Interpretation when omitted
  The file is output to the current folder.
  The output object file name will be the source file name with the extension replaced by ".obj".

[Detailed description]

- This option specifies the object file name as *file*.
- If *file* already exists, it will be overwritten.
- Even if this option is specified, when a fatal error occurs, the object file cannot be output.
- An error will occur if two or more files are output.
- An error will occur if *file* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Individual Assemble Options] tab, [Object file name] in the [Output File] category

[Example of use]

- To output the object file with "sample.obj" as the file name, describe as:

```
>asrh -osample.obj -Xcommon=v850e3v5 main.asm
```

-Xobj_path

This option specifies the folder to save an object file generated during assembling.

[Specification format]

```
-Xobj_path[=path]
```

- Interpretation when omitted
  The object file is saved under the source file name with the extension replaced by ".obj" to the current folder.

[Detailed description]

- This option specifies the folder to save an object file generated during assembling as *path*.
- If an existing folder is specified as *path*, the object file is saved under the source file name with the extension replaced by ".obj" to *path*.
  An error will occur if a nonexistent folder is specified.
- An existing file can be specified as *path*.
  If one object file is output, it will be saved with *path* as the file name.
  If two or more object files are output, an error will occur.
  An error will occur if a nonexistent file is specified.
- If "=*path*" is omitted, the object file is saved under the C source file name with the extension replaced by ".obj".
- If two or more files with the same name (even if they are in different folders) are specified as source files, then a warning is output, and an object file is only saved for the last source file to be specified.

[Example of use]

- To save the object file generated during assembling to foler "D:\sample", describe as:

```
>asrh -Xobj_path=D:\sample -Xcommon=v850e3v5 main.asm
```

-Xprn_path

This option specifies the folder to save the assemble list file.

[Specification format]

```
-Xprn_path[=path]
```

- Interpretation when omitted
  An assemble list file will not be output.

[Detailed description]

- This option specifies the folder to save the assemble list file output during assembling as *path*.
- If an existing folder is specified as *path*, the assemble list file is saved under the source file name with the extension replaced by ".prn" to *path*.
  An error will occur if a nonexistent folder is specified.
- An existing file can be specified as *path*.
- The assemble list file is saved with *path* as the file name.
  An error will occur if a nonexistent file is specified.
- If "=*path*" is omitted, the assemble list file is saved under the source file name with the extension replaced by ".prn".
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Output assemble list file], [Output folder for assemble list file] in the [Assemble List] category
  - From the [Assemble Options] tab, [Output assemble list file], [Output folder for assemble list file] in the [Assemble List] category
  - From the [Individual Compile Options] tab, [Output assemble list file], [Output folder for assemble list file] in the [Assemble List] category
  - From the [Individual Assemble Options] tab, [Output assemble list file], [Output folder for assemble list file] in the [Assemble List] category

[Example of use]

- To save the assemble list file output during assembling to foler "D:\sample", describe as:

```
>asrh -Xprn_path=D:\sample -Xcommon=v850e3v5 main.asm
```

Source debugging control

The source debugging control options are as follows.

- -g

-g

This option outputs information for source debugging.

[Specification format]

```
-g
```

- Interpretation when omitted
  Information for source debugging will not be output.

[Detailed description]

- This option outputs information for source debugging to the output file.
- Source debugging can be performed by specifying this option.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Assemble Options] tab, [Add debug information] in the [Debug Information] category
  - From the [Individual Assemble Options] tab, [Add debug information] in the [Debug Information] category

[Example of use]

- To output information for source debugging to the output file, describe as:

```
>asrh -g -Xcommon=v850e3v5 main.asm
```

Device specification

The device specification options are as follows.

- -Xcommon
- -Xcpu

-Xcommon

This option specifies that an object file common to the various devices is generated.

[Specification format]

```
-Xcommon=series
```

- Interpretation when omitted
  An error will occur.
  However, an error will not occur if the -V or -h option is specified.

[Detailed description]

- This option specifies that an object file common to the various devices is generated.
- When this option is specified, only instructions are used in the instruction set architecture of the target, and common magic number *series* that supports the instruction set architecture is embed into the object file.
- The item that can be specified as *series* is shown below.
  An abort error will occur if any other item is specified.

v850e3v5	It is possible to link to the model with the V850E3v5 instruction set architecture, which is specified as a target device.
rh850	The result is the same as when "v850e3v5" is specified.

- An error will occur if *series* is omitted.

[Example of use]

- To embed the magic number common to models with the instruction set architectures superior to V850E3V5 into the object file to be generated, describe as:

```
>asrh -Xcommon=v850e3v5 -c main.asm
```

-Xcpu

This option specifies that an object for the specified core is generated.

[Specification format]

```
-Xcpu=core
```

- Interpretation when omitted
  If the -Xcommon option is specified, an object for G3M is generated.

[Detailed description]

- This option specifies that an object for core *core* is generated.
- The items that can be specified as *core* are shown below.
  An abort error will occur if any other item is specified.

g3m	Generates an object for G3M.
g3k	Generates an object for G3K.

- An error will occur if *core* is omitted.

[Example of use]

- To generate an object for G3M, describe as:

```
>asrh -Xcpu=g3m -c -Xcommon=v850e3v5 main.asm
```

Preprocessor control

The preprocessor control options are as follows.

- -D
- -U
- -I

-D

This option defines assembler symbols.

[Specification format]

```
-Dname[=def][name[=def]]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option defines *name* as an assembler symbol.
- This is equivalent to adding "*name* .SET def" at the beginning of the assembly source program.
- If *name* contains characters that are allowed in an assembler symbol, but which cannot be used in a preprocessor macro ("@", ".", and "~"), a warning will be output, and it is defined as an assembler symbol only.
- An error will occur if *name* is omitted.
- If "=*def*" is omitted, *def* is regarded as 1.
- This option can be specified more than once.
- If both this option and -U option are specified for the same assembler symbol, the option specified last will be valid.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Macro definition] in the [Frequently Used Options(for Assemble)] category
  - From the [Assemble Options] tab, [Macro definition] in the [Preprocess] category
  - From the [Individual Assemble Options] tab, [Macro definition] in the [Preprocess] category

[Example of use]

- To define "sample=256" as an assembler symbol, describe as:

```
>asrh -Dsample=256 -Xcommon=v850e3v5 main.asm
```

-U

This option deletes the assembler symbol definition by the -D option.

[Specification format]

```
-Uname[,name]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option deletes the definition of assembler symbol *name* by the -D option.
- An error will occur if *name* is omitted.
- This option cannot delete the definition by describing "*name* .SET *def*".
- This option can be specified more than once.
- If both this option and -D option are specified for the same assembler symbol, the option specified last will be valid.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Assemble Options] tab, [Macro definition] in the [Preprocess] category
  - From the [Individual Assemble Options] tab, [Macro definition] in the [Preprocess] category

[Example of use]

- To delete the definition of assembler symbol "test" by the -D option, describe as:

```
>asrh -Utest -Xcommon=v850e3v5 main.asm
```

-I

This option specifies the folder to search include files.

[Specification format]

```
-Ipath[,path]...
```

- Interpretation when omitted
  The include file is searched from the standard include file folder.

[Detailed description]

- This option specifies the folder to search include files that are read by assembler control instruction "$INCLUDE/$BINCLUDE" as *path*.
  Include files are searched according to the following sequence.

(1) Path specified by the -I option

(2) Folder with source file

(3) Folder of original C source file

(4) Current folder

- If *path* does not exist, a warning will be output.
- An error will occur if *path* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Additional include paths], [System include paths] in the [Frequently Used Options(for Assemble)] category
  - From the [Assemble Options] tab, [Additional include paths], [System include paths] in the [Preprocess] category
  - From the [Individual Assemble Options] tab, [Additional include paths], [Use whole include paths specified for build tool] in the [Preprocess] category

[Example of use]

- To search include files from folder "D:\include", "D:\src", and the current folder in that order, describe as:

```
>asrh -ID:\include -Xcommon=v850e3v5 D:\src\main.asm
```

Japanese/Chinese character control

The Japanese/Chinese character control option is as follows.

- -Xcharacter_set

-Xcharacter_set

This option specifies the Japanese/Chinese character code.

[Specification format]

```
-Xcharacter_set=code
```

- Interpretation when omitted
  Processing of Japanese/Chinese character encoding is not performed.

[Detailed description]

- This option specifies the character code to be used for Japanese/Chinese comments and character strings in the source file.
- The items that can be specified as *code* are shown below.
  An error will occur if any other item is specified.
  Operation is not guaranteed if the specified character code differs from the character code of the source file.

none	Does not process the Japanese and Chinese character code
euc_jp	EUC (Japanese)
sjis	SJIS
utf8	UTF-8
big5	Traditional Chinese
gb2312	Simplified Chinese

- An error will occur if *code* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Assemble Options] tab, [Character encoding] in the [Character Encoding] category
  - From the [Individual Assemble Options] tab, [Character encoding] in the [Character Encoding] category

[Example of use]

- To specify EUC as the character code to be used for Japanese comments and character strings in the input file, describe as:

```
>asrh -Xcharacter_set=euc_jp -Xcommon=v850e3v5 main.asm
```

Generated code control

The generated code control options are as follows.

- -Xreg_mode
- -Xreserve_r2
- -Xep

-Xreg_mode

This option specifies the register mode.

[Specification format]

```
-Xreg_mode=mode
```

- Interpretation when omitted
  The 32-register mode object file is generated.

[Detailed description]

- This option generates the object file for the specified register mode.
- This option limits the number of registers used by ccrh to 32 (the 32-register mode) or 22 (the 22-register mode or register mode "common") and embeds the magic number into the object file.
- Use register mode "common" to generate the object file that does not depend on register modes.
- The items that can be specified as *mode* are shown below.
  An error will occur if any other item is specified.

Register Mode (*mode*)	Working Registers	Registers for Register Variables
common	r10 to r14	r25 to r29
22	r10 to r14	r25 to r29
32	r10 to r19	r20 to r29

- An error will occur if *mode* is omitted.
- An abort error will occur if the register that cannot be used for the assembly source file.
- The same register mode must be specified for all source files.
  The different register mode cannot be specified for each source file.
  If there are object files with different register modes, an error will occur during linking.
- By specifying this option, the register mode of the software register bank function can be changed.

[Example of use]

- To generate the 22-register mode object file, describe as:

```
>asrh -Xreg_mode=22 -Xcommon=v850e3v5 main.asm
```

-Xreserve_r2

This option reserves the r2 register.

[Specification format]

```
-Xreserve_r2
```

- Interpretation when omitted
  The compiler uses the r2 register without reserving it.

[Detailed description]

- This option reserves the r2 register and generates code that does not use this register by the compiler.

[Example of use]

- To reserve the r2 register and generates code that does not use this register by the compiler, describe as:

```
>asrh -Xreserve_r2 -Xcommon=v850e3v5 main.asm
```

-Xep

This option specifies how to handle the ep register.

[Specification format]

```
-Xep=mode
```

- Interpretation when omitted
  The ep register is treated as a register guaranteeing the value before and after the function call.

[Detailed description]

- This option specifies how to handle the ep register.
- The items that can be specified as *mode* are shown below.
  An error will occur if any other item is specified.

fix	Fixes the value of the ep register for the entire project. Specify this parameter when any of the following attribute strings in a #pragma section in the project is used. ep_auto, ep_disp4, ep_disp5, ep_disp7, ep_disp8, ep_disp16, ep_disp23
callee	Treats the ep register as a register guaranteeing the value before and after the function call. Specify this parameter when the -Omap or -Osmap option is specified.

- An error will occur if *mode* is omitted.

[Example of use]

- To fix the value of the ep register for the entire project, describe as:

```
>asrh -Xep=fix -Xcommon=v850e3v5 main.asm
```

Assembler control specification

The assembler control specification option is as follows.

- -Xasm_far_jump

-Xasm_far_jump

This option controls outputting far jump for an assembly source file.

[Specification format]

```
-Xasm_far_jump
```

- Interpretation when omitted
  Assembly is performed as a jarl or jr instruction.

[Detailed description]

- For an assembly source file, this option assumes that all jarl and jr instructions described in the source are jarl32 and jr32 instructions, and assembling is performed.
- If you wish to control individual instructions, add jarl22/jarl32 or jr22/jarl32 to the source.
- This option does not affect the jump instruction.
- If this option is specified for a C source file, that will be ignored without outputting a warning.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Compile Options] tab, [Use 32-bit branch instruction] in the [Output Code] category
  - From the [Assemble Options] tab, [Use 32-bit branch instruction] in the [Output Code] category
  - From the [Individual Compile Options] tab, [Use 32-bit branch instruction] in the [Output Code] category
  - From the [Individual Assemble Options] tab, [Use 32-bit branch instruction] in the [Output Code] category

[Example of use]

- To assume that all jarl and jr instructions described in the source are jarl32 and jr32 instructions, and perform assembling, describe as:

```
>asrh -Xasm_far_jump -Xcommon=v850e3v5 main.asm
```

Error output control

The error output control option is as follows.

- -Xerror_file

-Xerror_file

This option outputs error messages to a file.

[Specification format]

```
-Xerror_file=file
```

- Interpretation when omitted
  Error messages are output to only the standard error output.

[Detailed description]

- This option outputs error messages to the standard error output and file *file*.
- If *file* already exists, it will be overwritten.
- An error will occur if *file* is omitted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Individual Assemble Options] tab, [Output error message file], [Error message file output folder], [Error message file name] in the [Error Output] category

[Example of use]

- To output error messages to the standard error output and file "err", describe as:

```
>asrh -Xerror_file=err -Xcommon=v850e3v5 main.asm
```

Warning message output control

The warning message output control options are as follows.

- -Xno_warning

-Xno_warning

This option suppresses outputting warning messages of the specified number.

[Specification format]

```
-Xno_waning=num[,num]...
-Xno_waning=num1-num2
```

- Interpretation when omitted
  All warning messages are output.

[Detailed description]

- This option suppresses outputting warning messages of the specified number.
- Specify the error numbers as *num*, *num1*, and *num2*.
  If the error number that does not exist, it will be ignored.
- An error will occur if *num*, *num1*, or *num2* is omitted.
- If *num1-num2* is specified, it is assumed that error numbers within the range have been specified.
- The error number specified by this option is the rightmost 5 digits of the 7-digit number following the "W".
  See "CubeSuite+ Integrated Development Environment User's Manual: Message" for error numbers.
- This option controls the warning messages of the compiler and assembler and does not control the messages of the optimizing linker.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Individual Assemble Options] tab, [Undisplayed warning message] in the [Warning Message] category

[Example of use]

- To suppress outputting warning message "W0550002" and "W0550003", describe as:

```
>asrh -Xno_waning=50002,50003 -Xcommon=v850e3v5 main.asm
```

Subcommand file specification

The subcommand file specification option is as follows.

- @

@

This option specifies a subcommand file.

[Specification format]

```
@file
```

- Interpretation when omitted
  Only the options and file names specified on the command line are recognized.

[Detailed description]

- This option handles *file* as a subcommand file.
- An error will occur if *file* does not exist.
- An error will occur if *file* is omitted.
- See "B.4.2 Subcommand file usage" for details about a subcommand file.

[Example of use]

- To handle "command.txt" as a subcommand file, describe as:

```
>asrh @command.txt -Xcommon=v850e3v5
```

B.5.3 Link options

This section explains options for the link phase.

Caution about options are shown below.

- Uppercase characters and lowercase characters are not distinguished for options.
- Uppercase characters in options and parameters indicate that they can be specified as abbreviations for options and parameters.
  The characters after the uppercase characters can be omitted.

Example For example, -FOrm=Absolute can be specified as follows.
-fo=a
-fo=abs
-for=absolu

- When a file name is specified as a parameter, "(" and ")" cannot be used.
- When link options are specified for the ccrh command, the -Xlk_option option must be used.

The types and explanations for options are shown below.

Table B-4. Link Options

Classification	Option	Description
Input control	-Input	This option specifies the input file.
	-LIBrary	This option specifies the input library file.
	-Binary	This option specifies the input binary file.
	-DEFine	This option defines an undefined symbol forcedly.
	-ENTry	This option specifies the execution start address.
Output control	-FOrm	This option specifies the output format.
	-DEBug	This option outputs debug information to the output file.
	-NODEBug	This option does not output the debug information.
	-RECord	This option specifies the size of the data record to be output.
	-ROm	This option specifies the section that maps symbols from ROM to RAM.
	-OUtput	This option specifies the output file.
	-MAp	This option outputs the external variable allocation information file.
	-SPace	This option fills the vacant area of the output range.
	-Message	This option output information messages.
	-NOMessage	This option suppresses the output of information messages.
	-MSg_unused	This option notifies the user of the external defined symbol that is not referenced.
	-BYte_count	This option specifies the maximum byte count for a data record.
	-PADDING	This option fills in data at the end of a section.
	-OVERRUN_FETCH	This option prevents reading of vacant areas due to overrun fetch.
List output	-LISt	This option outputs the list file.
	-SHow	This option specifies information that is output to the list file.

Classification	Option	Description
Section specification	-STARt	This option specifies the start address of the section.
	-FSymbol	This option outputs external defined symbols to the symbol address file.
	-ALIGNED_SECTION	This option changes the number of alignment of the section to 16 bytes.
Verify specification	-CPu	This option checks the consistency of the address to which the section is allocated.
Subcommand file specification	-SUbcommand	This option specifies options with a subcommand file.
Other	-S9	This option outputs the S9 record at the end.
	-STACk	This option outputs the stack information file.
	-COmpress	This option compresses the debug information.
	-NOCOmpress	This option does not compress the debug information.
	-MEMory	This option specifies the memory size occupied during linking.
	-REName	This option changes an external symbol name or a section name.
	-DELete	This option deletes an external symbol name or a library module.
	-REPlace	This option replaces library modules.
	-EXTract	This option extracts library modules.
	-STRip	This option deletes debug information in the load module file or library file.
	-CHange_message	This option changes the type of information, warning, and error messages.
	-Hide	This option deletes local symbol name information from the output file.
	-Total_size	This option displays the total size of sections after the linking to the standard error output.
	-LOgo	This option outputs the copyright notice.
	-NOLOgo	This option suppresses the output of the copyright notice.
	-END	This option executes option strings specified before this option.
	-EXIt	This option specifies the end of option specifications.

Input control

The input control options are as follows.

- -Input
- -LIBrary
- -Binary
- -DEFine
- -ENTry

-Input

This option specifies the input file.

[Specification format]

```
-Input=file[(module[,module]...)][{,|Δ}file[(module[,module]...)]]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option specifies input file *file*.
  If multiple files are specified, delimit them with a comma (,) or space.
- Wildcard characters (*, ?) can also be used.
  The character strings specified with wildcard characters are expanded in alphabetical order.
  Expansion of numerical values precedes that of alphabetic characters. Uppercase characters are expanded before lowercase characters.
- Files that can be specified as input files are object files output from the compiler or the assembler and relocatable files, load module files, Intel HEX files, and Motorola S-record files output from the optimizing linker.
  In addition, a module in a library can be specified using the format of "*library*(*module*)".
  Specify the module name without the extension.
- If no extension is specified for the input filename, then if no module name is specified, it is assumed to be ".obj"; if a module name is specified, it is assumed to be ".lib".
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Object file] in the [Input File] category
  - From the [Create Library Options] tab, [Object file] in the [Input File] category

[Caution]

- This option can be used only in a subcommand file.
  An error will occur if this option is specified on the command line.
  When input files are specified on the command line, specify them without the -input option.

[Example of use]

- To input a.obj and module "e" in lib1.lib, describe as:
  <Command line>

```
>rlink -subcommand=sub.txt
```

<Subcommand file "sub.txt">

```
-input=a.obj lib1(e)
```

- To input all ".obj" files beginning with "c", describe as:
  <Command line>

```
>rlink -subcommand=sub.txt
```

<Subcommand file "sub.txt">

```
-input=c*.obj
```

[Remark]

- If the -form=object or -extract option is specified, this option will be invalid.
- If an Intel HEX file is specified as an input file, only the -form=hexadecimal option can be specified. If a Motorola S-record file is specified, only the -form=stype option can be specified.
  If the output file name is not specified, it will be "*first input file name*_combine.*extension*" (If the input file is "a.mot", the output file will be "a_combine.mot").

-LIBrary

This option specifies the input library file.

[Specification format]

```
-LIBrary=file[,file]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option specifies input library file *file*.
  If multiple files are specified, delimit them with a comma (,).
- Wildcard characters (*, ?) can also be used.
  The character strings specified with wildcard characters are expanded in alphabetical order.
  Expansion of numerical values precedes that of alphabetic characters. Uppercase characters are expanded before lowercase characters.
- If the extension is omitted from the input file specification, it is assumed that ".lib" has been specified.
- If this option and the -form=library or -extract option are specified at the same time, the specified library file is input as the target library to be edited.
  Otherwise, undefined symbols are searched in the library file after the link processing between files specified as the input files are executed.
- The symbols are searched in the library file in the following sequence:
  - User library files specified by this option (in the specified order)
  - System library files specified by this option (in the specified order)
  - Default library (environment variables "HLNK_LIBRARY1", "HLNK_LIBRARY2", and "HLNK_LIBRARY3"[Note] in that order)

Note See "B.3 Environment Variable" for details about environment variables.

- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Using libraries], [Use standard libraries], [Use "Standard Library" function], [Use "Mathematical Library (Double precision)" function], [Use "Mathematical Library (Single precision)" function], [Use "Non-local jump Library" function] in the [Frequently Used Options(for Link)] category, [Use standard libraries], [Use "Standard Library" function], [Use "Mathematical Library (Double precision)" function], [Use "Mathematical Library (Single precision)" function], [Use "Non-local jump Library" function] in the [Frequently Used Options(for Create Library)] category
  - From the [Link Options] tab, [Using libraries], [System libraries], [Use standard libraries], [Use "Standard Library" function], [Use "Mathematical Library (Double precision)" function], [Use "Mathematical Library (Single precision)" function], [Use "Non-local jump Library" function] in the [Library] category
  - From the [Create Library Options] tab, [Using libraries], [System libraries], [Use standard libraries], [Use "Standard Library" function], [Use "Mathematical Library (Double precision)" function], [Use "Mathematical Library (Single precision)" function], [Use "Non-local jump Library" function] in the [Library] category

[Example of use]

- To input a.lib and b.lib, describe as:

```
rlink main.obj -library=a.lib,b
```

- To input all ".lib" files beginning with "c", describe as:

```
rlink main.obj -library=c*.lib
```

-Binary

This option specifies the input binary file.

[Specification format]

```
-Binary=file(section[:alignment][/attribute][,symbol])[,file(section[:alignment][/attribute]
[,symbol])]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option specifies input binary file *file*.
  If multiple files are specified, delimit them with a comma (,).
- If the extension is omitted from the input file specification, it is assumed that ".bin" has been specified.
- Input binary data is allocated as the data of specified section *section*.
  Specify the section address by the -start option.
  An error will occur if *section* is omitted.
- When symbol *symbol* is specified, it can be linked as a defined symbol.
  For a variable name referenced by a C program, add "_" at the head of the reference name in the program.
- The section specified by this option can have its section attribute and number of alignment specified.
- CODE or DATA can be specified as section attribute *attribute*.
  If *attribute* is omitted, the write, read, and execute attributes will be all valid by default.
- The value that can be specified for number of alignment *alignment* is a power of 2 (1, 2, 4, 8, 16, or 32).
  Other value cannot be specified.
  If *alignment* is omitted, "1" will be valid by default.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Binary file] in the [Input File] category
  - From the [Create Library Options] tab, [Binary file] in the [Input File] category

[Example of use]

- b.bin is allocated from 0x200 as the D1bin section.
  c.bin is allocated after D1bin as the D2bin section (with the number of alignment = 4).
  The c.bin data is linked as defined symbol "_datab".
  To perform the above operations, describe as:

```
>rlink a.obj -start=D*/200 -binary=b.bin(D1bin),c.bin(D2bin:4,_datab)
```

[Remark]

- If the -form={object|library} option or -strip option is specified, this option will be invalid.
- If input object file is not specified, this option cannot be specified.

-DEFine

This option defines an undefined symbol forcedly.

[Specification format]

```
-DEFine=symbol1={symbol2|value}[,symbol1={symbol2|value}]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option defines undefined symbol *symbol1* forcedly as external defined symbol *symbol2* or numerical value *value*.
- Specify *value* in hexadecimal.
  If the specified value starts with a character from A to F, symbols are searched first, and if corresponding symbol is not found, the value is interpreted as a numerical value.
  Values starting with 0 are always interpreted as numerical values.
- If the specified symbol name is a C variable name, add "_" at the head of the definition name in the program.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Symbol definition] in the [Input File] category

[Example of use]

- To define "_sym1" as the same value as external defined symbol "data", describe as:

```
>rlink -define=_sym1=data a.obj b.obj
```

- To define "_sym2" as 0x4000, describe as:

```
>rlink -define=_sym2=4000 a.obj b.obj
```

[Remark]

- If the -form={object|relocate|library} option is specified, this option will be invalid.

-ENTry

This option specifies the execution start address.

[Specification format]

```
-ENTry={symbol|address}
```

- Interpretation when omitted
  None

[Detailed description]

- This option defines execution start address with external defined symbol *symbol* or address *address*.
- Specify *address* in hexadecimal.
  If the specified value starts with a character from A to F, defined symbols are searched first, and if corresponding symbol is not found, the value is interpreted as an address.
  Values starting with 0 are always interpreted as addresses.
- If the specified symbol name is a C variable name, add "_" at the head of the definition name in the program.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Specify execution start address], [Execution start address] in the [Output Code] category

[Example of use]

- To specify main function in C as the execution start address, describe as:

```
>rlink -entry=_main a.obj b.obj
```

- To specify 0x100 as the execution start address, describe as:

```
>rlink -entry=100 a.obj b.obj
```

[Remark]

- If the -form={object|relocate|library} option or -strip option is specified, this option will be invalid.

Output control

The output control options are as follows.

- -FOrm
- -DEBug
- -NODEBug
- -RECord
- -ROm
- -OUtput
- -MAp
- -SPace
- -Message
- -NOMessage
- -MSg_unused
- -BYte_count
- -PADDING
- -OVERRUN_FETCH

-FOrm

This option specifies the output format.

[Specification format]

```
-FOrm={format}
```

- Interpretation when omitted
  A load module file is output (It is the same result as when the -form=absolute option is specified).

[Detailed description]

- This option specifies output format *format*.
- The items that can be specified as *format* are shown below.

Absolute	Outputs a load module file.
Relocate	Outputs a relocatable file.
Object	Outputs an object file. Use this when a module is extracted as an object file from a library by the -extract option.
Library[={S|U}]	Outputs a library file. When "library=s" is specified, a system library file is output. When "library=u" is specified, a user library file is output. If only "library" is specified, it is assumed that "library=u" has been specified.
Hexadecimal	Outputs an Intel HEX file. See "3.5 Intel HEX File" for details.
Stype	Outputs a Motorola S-record file. See "3.6 Motorola S-record File" for details.
Binary	Outputs a binary file.

- If *format* is omitted, it is assumed that "Absolute" has been specified.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Output hex file], [Hex file format] in the [Frequently Used Options(for Hex Output)] category, [Output file format] in the [Frequently Used Options(for Create Library)] category
  - From the [Hex Output Options] tab, [Output hex file], [Use object uniting function] in the [Output File] category, [Hex file format] in the [Hex Format] category
  - From the [Create Library Options] tab, [Output file format] in the [Output File] category

[Remark]

- The relations between output formats and input files or other options are shown below.

Table B-5. Relations Between Output Formats And Input Files Or Other Options

Output Format	Specified Option	File Format That Can Be Input	Specifiable Option Note 1
Absolute	-strip specified	Load module file	-input, -output
	Other than above	Object file Relocatable file Binary file Library file	-input, -library, -binary, -debug, -nodebug, -cpu, -start, -rom, -entry, -output, -map, -hide, -compress, -rename, -delete, -define, -fsymbol, -stack, -memory, -msg_unused, -show={symbol|reference|xreference}, -aligned_section, -overrun_fetch
Relocate	-extract specified	Library file	-library, -output
	Other than above	Object file Relocatable file Binary file Library file	-input, -library, -debug, -nodebug, -output, -hide, -rename, -delete, -show={symbol|xreference}
Object	-extract specified	Library file	-library, -output
Hexadecimal Stype Binary		Object file Relocatable file Binary file Library file	-input, -library, -binary, -cpu, -start, -rom, -entry, -output, -map, -space, -rename, -delete, -define, -fsymbol, -stack, -record, -s9Note 2, -byte_countNote 3, -memory, -msg_unused, -show={symbol|reference|xreference}, -aligned_section, -overrun_fetchNote 4
		Load module file	-input, -output, -record, -s9Note 2, -byte_countNote 3, -show={symbol|reference|xreference}
		Intel HEX fileNote 5	-input, -output
		Motorola S-record fileNote 5	-input, -output, -s9Note 2
Library	-strip specified	Library file	-library, -output, -memoryNote 6, -show={symbol|section}
	-extract specified	Library file	-library, -output
	Other than above	Object file Relocatable file	-input, -library, -output, -hide, -rename, -delete, -replace, -memoryNote 6, -show={symbol|section}

Notes 1. The following options can always be specified.
-message, -nomessage, -change_message, -logo, -nologo, -form, -list, -subcommand

2. The -s9 option is valid only when the -form=stype option is specified.

3. The -byte_count option is valid only when the -form= hexadecimal option is specified.

4. The -overrun_fetch option is valid only when the -form=hexadecimal or -form=stype option is specified.

5. If an Intel HEX file is specified as an input file, only the -form=hexadecimal option can be specified. If a Motorola S-record file is specified, only the -form=stype option can be specified.
6. The -memory option cannot be specified when the -hide option is specified.

[Example of use]

- To output relocatable file c.rel from a.obj and b.obj, describe as:

```
>rlink a.obj b.obj -form=relocate -output=c.rel
```

- To extract module "a" from lib.lib and output as an object file, describe as:

```
>rlink -library=lib.lib -extract=a -form=object
```

- To extract module "a" from lib.lib and output library file exta.lib, describe as:

```
>rlink -library=lib.lib -extract=a -form=library -output=exta
```

- To extract module "a" from lib.lib and output relocatable file a.rel, describe as:

```
>rlink -library=lib.lib -extract=a -form=relocate
```

-DEBug

This option outputs debug information to the output file.

[Specification format]

```
-DEBug
```

- Interpretation when omitted
  The debug information is output to the output file (It is the same result as when the -debug option is specified).

[Detailed description]

- This option outputs debug information to the output file.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Output debug information] in the [Debug Information] category
  - From the [Create Library Options] tab, [Output debug information] in the [Debug Information] category

[Example of use]

- To output debug information to the output file, describe as:

```
>rlink a.obj b.obj -debug -output=c.abs
```

[Remark]

- If the -form={object|library|hexadecimal|stype|binary}, -strip option or -extract option is specified, this option will be invalid.
- If two or more output file names are specified using the -form=absolute option and -output option, the debug information will not be output.

-NODEBug

This option does not output the debug information.

[Specification format]

```
-NODEBug
```

- Interpretation when omitted
  The debug information is output to the output file (It is the same result as when the -debug option is specified).

[Detailed description]

- This option does not output the debug information.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Output debug information] in the [Debug Information] category
  - From the [Create Library Options] tab, [Output debug information] in the [Debug Information] category

[Example of use]

- Not to output the debug information, describe as:

```
>rlink a.obj b.obj -nodebug -output=c.abs
```

[Remark]

- If the -form={object|library|hexadecimal|stype|binary}, -strip option or -extract option is specified, this option will be invalid.

-RECord

This option specifies the size of the data record to be output.

[Specification format]

```
-RECord={record}
```

- Interpretation when omitted
  Various data records are output according to each address.

[Detailed description]

- This option outputs data with data record *record* regardless of the address range.
- The items that can be specified as *record* are shown below.

H16	HEX record
H20	Expanded HEX record
H32	32-bit HEX record
S1	S1 record
S2	S2 record
S3	S3 record

- If there is an address that is larger than the specified data record, the appropriate data record is selected for the address.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Hex Output Options] tab, [Unify record size], [Unify record size] in the [Hex Format] category

[Example of use]

- To output 32-bit HEX record regardless of the address range:

```
>rlink a.obj b.obj -record=H32 -form=hexadecimal -output=c.abs
```

[Remark]

- If the -form={hexadecimal|stype} option is not specified, this option will be invalid.
- An error will occur if the -record={S1|S2|S3} option is specified when the -form=hexadecimal option is specified, or if the -record={H16|H20|H32} option is specified when the -form=stype option is specified.

-ROm

This option specifies the section that maps symbols from ROM to RAM.

[Specification format]

```
-ROm=ROMsection=RAMsection[,ROMsection=RAMsection]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option reserves ROM and RAM areas in the initialized data area and relocates defined symbols in the ROM section with the address in the RAM section.
- Specify a relocatable section including the initial value for ROM section *ROMsection*.
- Specify a a nonexistent section or relocatable section whose size is 0 for RAM section *RAMsection*.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [ROM to RAM mapped section] in the [Section] category

[Example of use]

- To reserve the .data.R section with the same size as the .data section and relocate defined symbols in the .data section with address in the .data.R section, describe as:

```
>rlink a.obj b.obj -rom=.data=.data.R -start=.data/100,.data.R/8000
```

[Remark]

- If the -form={object|relocate|library} option or -strip option is specified, this option will be invalid.

-OUtput

This option specifies the output file.

[Specification format]

```
-OUtput=file[={address1-address2|section[:section]...}][,file[={address1-address2|section
[:section]...}]]...
```

- Interpretation when omitted
  The output file name is "*first-input-file-name.default-extension*".
  The default extensions are shown below.
    When the -form=absolute option is specified: abs
    When the -form=relocate option is specified: rel
    When the -form=object option is specified: obj
    When the -form=library option is specified: lib
    When the -form=hexadecimal option is specified: hex
    When the -form=stype option is specified: mot
    When the -form=binary option is specified: bin

[Detailed description]

- This option specifies output file *file*.
- Specify the start address and end address of the output range in hexadecimal as *address1* and *address2*.
  The output range including "-" is always interpreted as addresses.
- Specify the section to be output as *section*.
  If multiple files are specified, delimit them with a colon (:).
- If this option and the -form={absolute|hexadecimal|stype|binary} option are specified at the same time, two or more files can be specified.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Common Options] tab, [Output folder], [Output file name] in the [Frequently Used Options(for Link)] category, [Output folder], [Output file name], [Division output file] in the [Frequently Used Options(for Hex Output)] category, [Output folder], [Output file name] in the [Frequently Used Options(for Create Library)] category
  - From the [Link Options] tab, [Output folder], [Output file name] in the [Output File] category
  - From the [Hex Output Options] tab, [Output folder], [Output file name], [Division output file], [Output folder for united hex file] in the [Output File] category
  - From the [Create Library Options] tab, [Output folder], [Output file name] in the [Output File] category

[Example of use]

- To output the range from 0 to 0xffff to file1.abs and the range from 0x10000 to 0x1ffff to file2.abs, describe as:

```
>rlink a.obj b.obj -output=file1.abs=0-ffff,file2.abs=10000-1ffff
```

- To output the sec1 and sec2 sections to file1.abs and the sec3 section to file2.abs, describe as:

```
>rlink a.obj b.obj -output=file1.abs=sec1:sec2,file2.abs=sec3
```

[Remark]

- If a input file is an Intel Hex file or Motorola S-record file, two or more output files cannot be specified by this option. If this option is omitted, the output file name will be "*first input file name*_combine.*extension*" (If the input file is "a.mot", the output file will be "a_combine.mot").

-MAp

This option outputs the external variable allocation information file.

[Specification format]

```
-MAp[=file]
```

- Interpretation when omitted
  None

[Detailed description]

- This option outputs external variable allocation information file *file* that is used by the compiler in optimizing access to external variables.
- If the specification of the file name is omitted, the file name is the one specified by the -output option or "*first-input file-name*.bls".
- If the order of the declaration of variables in the external variable allocation information file is not the same as the order of the declaration of variables found when the object was read after recompilation, an error will occur.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Output external symbol allocation information file] in the [Optimization(Details)] category

[Example of use]

- To output external variable allocation information file file.bls, describe as:

```
>rlink a.obj b.obj -output=c.abs -map=file.bls
```

[Remark]

- This option is valid only when the -form={absolute|hexadecimal|stype|binary} option is specified.

-SPace

This option fills the vacant area of the output range.

[Specification format]

```
-SPace[=data]
```

- Interpretation when omitted
  None

[Detailed description]

- This option fills the vacant area of the output range with user-specified data *data*.
- The items that can be specified as *data* are shown below.

Numerical Value	Hexadecimal value
Random	Random number

- The following vacant areas are filled with the value according to the output range specification in the -output option:
  - When the section is specified as the output range
    - The specified value is output to vacant areas between the specified sections.
  - When the start address and end address is specified as the output range
    - The specified value is output to vacant areas within the specified range.
- Output data sizes in units of 1, 2, or 4 bytes are valid. The size is determined by the hexadecimal number specified using this option.
  If a 3-byte value is specified, the upper digit is extended with 0 to handle it as a 4-byte value.
  If an odd number of digits is specified, the upper digit is extended with 0 to handle it as an even number of digits.
- If the size of a vacant area is not a multiple of the size of the output data, the value is output as many times as possible, and then a warning will be output.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Hex Output Options] tab, [Fill unused areas in the output ranges with the value], [Output padding data] in the [Hex Format] category

[Example of use]

- To fill the vacant memory area with "ffH" within the range from address 100H to address 2FFH, describe as:

```
>rlink a.obj b.obj -form=hexadecimal -output=file1=100-2ff -start=P1/100,P2/200 -space=ff
```

[Remark]

- If the specification of the data is omitted in this option, vacant areas are not filled with values.
- This option is valid only when the -form={binary|stype|hexadecimal} option is specified.
- If the output range is not specified in the -output option, this option will be invalid.

-Message

This option output information messages.

[Specification format]

```
-Message
```

- Interpretation when omitted
  The output of information messages is suppressed (It is the same result as when the -nomessage option is specified).

[Detailed description]

- This option output information messages.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Enable information message output] in the [Message] category
  - From the [Create Library Options] tab, [Enable information message output] in the [Message] category

[Example of use]

- To output information messages, describe as:

```
>rlink a.obj b.obj -message
```

-NOMessage

This option suppresses the output of information messages.

[Specification format]

```
-NOMessage[=num[-num][,num[-num]]...]
```

- Interpretation when omitted
  The output of information messages is suppressed.

[Detailed description]

- This option suppresses the output of information messages.
- If message number *num* is specified, the output of the message with the specified number is suppressed.
  Also, a range of message numbers can be specified using a hyphen (-).
- Specify the 4-digit number that is output after the component number (05) and the phase of occurrence (6) as *num* (for example, specify 0004 for message number M0560004).
  0 at the beginning of the 4-digit number can be omitted (for example, specify 4 for message number M0560004).
- If a number of a warning or error type message is specified, the output of the message is suppressed assuming that -change_message option has changed the specified message to the information type.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Enable information message output, [Suppress number of information message] in the [Message] category
  - From the [Create Library Options] tab, [Enable information message output], [Suppress number of information message] in the [Message] category

[Example of use]

- To suppress outputting messages of M0560004, M0560100 to M0560103, and M0560500, describe as:

```
>rlink a.obj b.obj -nomessage=4,100-103,500
```

-MSg_unused

This option notifies the user of the external defined symbol that is not referenced.

[Specification format]

```
-MSg_unused
```

- Interpretation when omitted
  None

[Detailed description]

- This option notifies the user of the external defined symbol that is not referenced during link processing through an output message.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Notify unused symbol] in the [Message] category

[Example of use]

- To notify the user of the external defined symbol that is not referenced, describe as:

```
>rlink a.obj b.obj -message -msg_unused
```

[Remark]

- If a load module file is input, this option will be invalid.
- This option must be specified together with the -message option.
- The a message may be output for the function that inline expansion was performed during compilation.
  In this case, add a static declaration for the function definition to suppress the output of the message.
- If there are references to constant symbols within the same file, references are not correctly analyzed so that information notified by the output messages will be incorrect.

-BYte_count

This option specifies the maximum byte count for a data record.

[Specification format]

```
-BYte_count=num
```

- Interpretation when omitted
  An Intel HEX file is generated assuming that the maximum byte count is "FF".

[Detailed description]

- This option specifies maximum byte count for a data record when a file *num* when an Intel HEX file.
- Specify a one-byte hexadecimal value (01 to FF) as *num*.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Hex Output Options] tab, [Specify byte count for data record], [Maximum byte count for data record] in the [Hex Format] category

[Example of use]

- To specify 10 as the maximum byte count for a data record, describe as:

```
>rlink a.obj b.obj -form=hexadecimal -byte_count=10
```

[Remark]

- When a generated file is not an Intel HEX file (when the -form=hexadecimal option is not specified), this option will be invalid.

-PADDING

This option fills in data at the end of a section.

[Specification format]

```
-PADDING
```

- Interpretation when omitted
  None

[Detailed description]

- This option fills in data at the end of a section so that the section size is a multiple of the alignment of the section.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Fill with padding data at the end of a section] in the [Output Code] category

[Example of use]

- In the following case, 2 bytes of padding data are filled in the P section, and linking is performed with a size of 0x08.
  Alignment of the P section: 4 bytes
  Size of the P section: 0x06 bytes
  Alignment of the C section: 1 byte
  Size of the C section: 0x03 bytes

```
>rlink a.obj b.obj -start=P,C/0 -padding
```

- In the following case, if 2 bytes of padding data are filled in the P section, and linking is performed with a size of 0x08, then an error will be output because it overlaps with the C section.
  Alignment of the P section: 4 bytes
  Size of the P section: 0x06 bytes
  Alignment of the C section: 1 byte
  Size of the C section: 0x03 bytes

```
>rlink a.obj b.obj -start=P/0,C/7 -padding
```

[Remark]

- The value of the generated padding data is 0x00.
- Since padding is not performed to an absolute address section, the size of an absolute address section should be adjusted by the user.

-OVERRUN_FETCH

This option prevents reading of vacant areas due to overrun fetch.

[Specification format]

```
-OVERRUN_FETCH
```

- Interpretation when omitted
  None

[Detailed description]

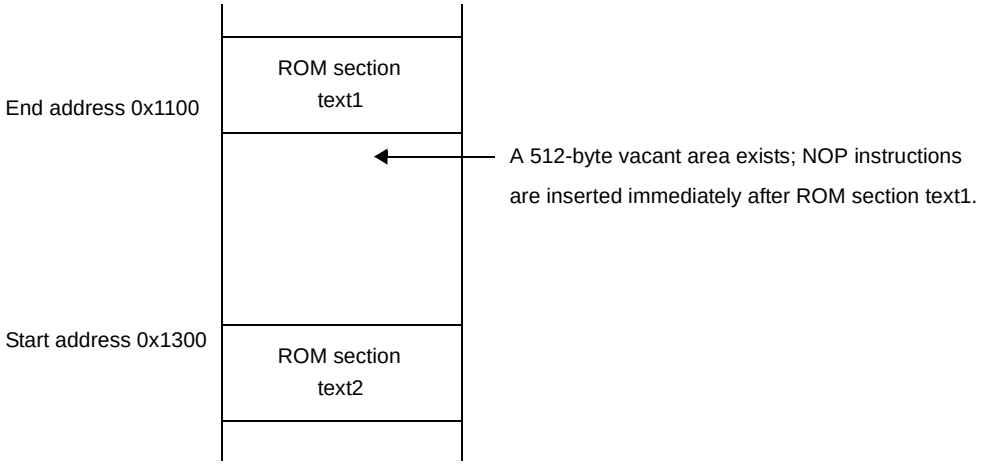
- 128-byte NOP instructions are inserted in a vacant area when a 128-byte or larger vacant area exists between sections or immediately after the end code section.
  NOP instructions are inserted according to the allocation specified by the -start option and output with the following section name obtained from the name of the section immediately before the inserted NOP instructions.

```
$sss_fetch??
```

sss: Name of the section immediately before the inserted NOP instructions
??: 0199

(1) Insert between sections

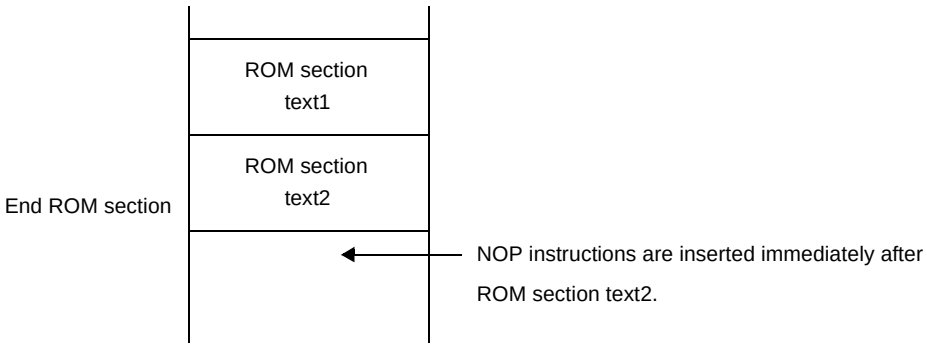
- NOP instructions are inserted when a 128-byte or larger vacant area exists between two ROM sections (executable program sections or sections (except program sections) allocated to the ROM area) or between a ROM section and a RAM section (a section allocated to the RAM area).



- For insertion between a ROM section and RAM section, NOP instructions are inserted only when the RAM section is allocated to larger addresses than the ROM section addresses (When the RAM section addresses are smaller than the ROM section addresses, NOP instructions are not inserted between them).

(2) Insert immediately after the end section

- NOP instructions are inserted immediately after the end ROM section.



  - When an address range is specified by the -cpu option, if the vacant area between the end address of the end section and the end address of the specified range is smaller than 128 bytes, NOP instructions will not be inserted.
  - If the vacant area between the end address of the end section and an 8-Mbyte address boundary is smaller than 128 bytes, NOP instructions will not be inserted.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Work around overrun fetch] in the [Output Code] category

[Example of use]

- To prevent reading of vacant areas due to overrun fetch, decsribe as:

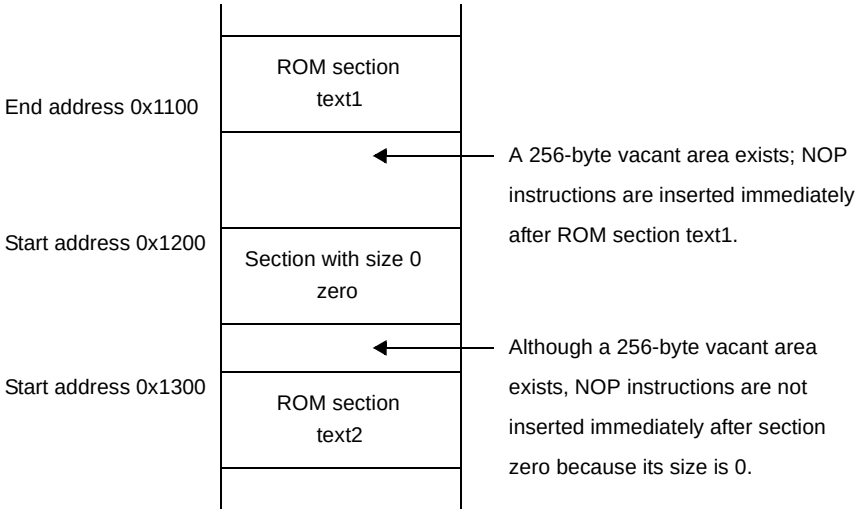
```
>rlink a.obj b.obj -overrun_fetch
```

[Remark]

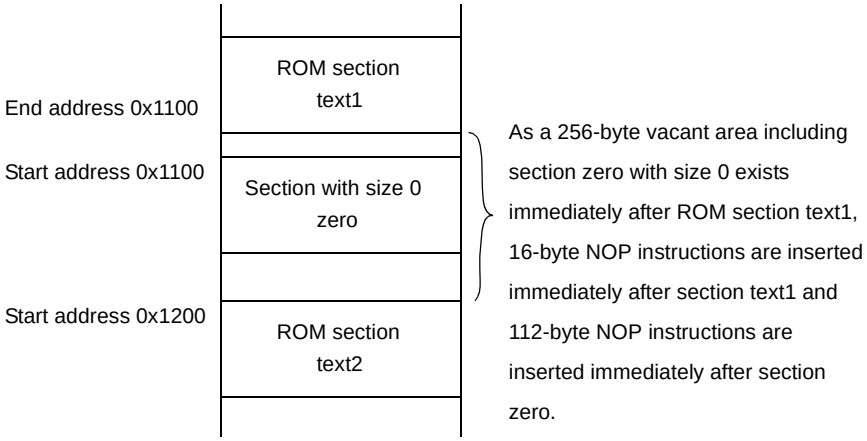
- When no address range is specified by the -cpu option and the end section is a ROM section, NOP instructions are always inserted immediately after the end ROM section.
- If a load module file is input, this option will be invalid.
- If the -form={object|relocate|library} option is specified, this option will be invalid.
- When overlaid sections are specified by the -start option, NOP instructions are not inserted for a string of sections specified as being overlaid by the -start option.

Examples 1. When option -start=A,B,E/400,C,D:F:G/8000 is specified, NOP instructions are not inserted between sections C, D, F, and G and after the end code section because overlaid sections are specified.

2. When a 128-byte or larger vacant area exists immediately after a section with size 0, NOP instructions are not inserted.



3. NOP instructions are inserted when a 128-byte or larger vacant area including a section with size 0 exists immediately after a ROM section.



List output

The list output options are as follows.

- -LISt
- -SHow

-LISt

This option outputs the list file.

[Specification format]

```
-LISt[=file]
```

- Interpretation when omitted
  None

[Detailed description]

- This option outputs list file *file*.
- An error will occur if the specification of the file name is omitted.

Specified Option	File Type	File Name
-form=library option or -extract option	Library list file	*Output file name*[Note].lbp
Other than above	Link map file	*Output file name*[Note].map

Note If there are two or more output files, this is the first input file name.

- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Output link map file] in the [List] category
  - From the [Create Library Options] tab, [Output link map file] in the [List] category

[Example of use]

- To output the link map file to file.map, describe as:

```
>rlink a.obj b.obj -list=file.map
```

-SHow

This option specifies information that is output to the list file.

[Specification format]

```
-SHow[=info[,info]...]
```

- Interpretation when omitted
  None

[Detailed description]

- This option specifies information *info* that is output to the list file.
- The items that can be specified as *info* are shown below.

Output Information (*info*)	When -form=library Option Is Specified	When Other Than -form=library Option Is Specified
SYmbol	Outputs symbol names within a module.	Outputs the symbol address, size, type, and optimization contents.
Reference	Not specifiable	Outputs the symbol address, size, type, optimization contents, and number of symbol references.
SEction	Outputs section names in a module.	Not specifiable
Xreference	Not specifiable	Outputs cross reference information.
Total_size	Not specifiable	Outputs the total sizes of sections separately for ROM-allocated sections and RAM-allocated sections.
ALL	Outputs symbol names and section names in a module.	When the -form=relocate option is specified Outputs the same information as when the -show=symbol,xreference,total_size option is specified. When the -form=absolute option is specified Outputs the same information as when the -show=symbol,reference,xreference,total_size option is specified. When the -form=hexadecimal/stype/binary option Outputs the same information as when the -show=symbol,reference,xreference,total_size option is specified. When the -form=object option is specified Not specifiable

Remark See "3.2 Link Map File" and "3.4 Library List File" for details about output information.

- See [Remark] for details about when the specification of output information is omitted.
- If the -extract option is specified, output information (*info*) cannot be specified.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Output link map file], [Output symbol information], [Output number of symbol reference], [Output cross reference information], [Output total sizes of sections] in the [List] category
  - From the [Create Library Options] tab, [Output link map file], [Output symbol information], [Output section list in a module], [Output cross reference information], [Output total sizes of sections] in the [List] category

[Example of use]

- To output the symbol address, size, type, optimization contents, and number of symbol references, describe as:

```
>rlink a.obj b.obj -list -show=symbol,reference
```

[Remark]

- The following table shows whether output information *info* will be valid or invalid by the combinations of the -form option and the -show or -show=all option.

		Symbol	Reference	Section	Xreference	Total_size
-form=absolute	only -show	Valid	Valid	Invalid	Invalid	Invalid
	-show=all	Valid	Valid	Invalid	Valid	Valid
-form=library	only -show	Valid	Invalid	Valid	Invalid	Invalid
	-show=all	Valid	Invalid	Valid	Invalid	Invalid
-form=relocate	only -show	Valid	Invalid	Invalid	Invalid	Invalid
	-show=all	Valid	Invalid	Invalid	Valid[Note 1]	Valid
-form=object	only -show	Valid	Valid	Invalid	Invalid	Invalid
	-show=all	Invalid	Invalid	Invalid	Invalid	Invalid
-form=hexadecimal[Note 2]/ stype[Note 3]/binary	only -show	Valid	Valid	Invalid	Invalid	Invalid
	-show=all	Valid	Valid	Invalid	Valid	Valid[Note 1]

Notes 1. If a load module file is input, this combination will be invalid.

2. If a input file is an Intel Hex file, the -show option cannot be specified.

3. If a input file is a Motorola S-record file, the -show option cannot be specified.

The limitations on the output of the cross reference information are shown below.

- When a load module file is input, the referrer address information is not output.
- The information about references to constant symbols within the same file is not output.
- When optimization is specified during compilation, information about branches to immediate subordinate functions is not output.
- Both -show=total_size option and -total_size option output the same information.

Section specification

The section specification options are as follows.

- -STARt
- -FSymbol
- -ALIGNED_SECTION

-STARt

This option specifies the start address of the section.

[Specification format]

```
-STARt=[(]section[{:|,}section]...[)][{:|,}section]...[/address][,[(]section[{:|,}section]...
[)][{:|,}section]...[/address]]...
```

- Interpretation when omitted
  Absolute address sections are allocated from smallest to largest, and then relative address sections starting at the end of the absolute address sections are allocated, in the order of appearance of the input files.

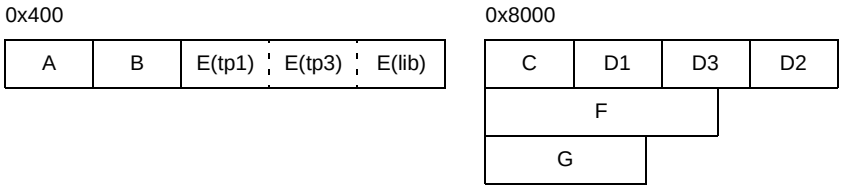
[Detailed description]

- This option specifies start address *address* of section *section*.
  Specify *address* in hexadecimal.
- Wildcard characters (*, ?) can also be used for *section*.
  The section specified with wildcard characters are expanded in the input order.
- Two or more sections (specifing by delimiting them with a comma (,)) can be allocated to the same address (i.e., sections are overlaid) by delimiting them with a colon (:).
  Sections specified at a single address are allocated in their specified order.
  Sections to be overlaid can be changed by enclosing them by parentheses "()".
- Objects in a single section are allocated in the specified order of the input file and the input library.
- If the specification of an address is omitted, the section is allocated from address 0.
- A section that is not specified by the -start option is allocated after the last allocation address.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Section start address] in the [Section] category

[Example of use]

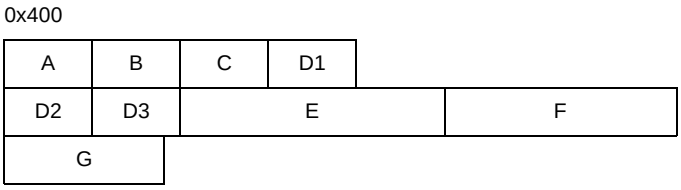
- The example below shows how sections are allocated when the objects are input in the following order (The names enclosed by parentheses are sections in each object).
  tp1.obj(A,D1,E)
  tp2.obj(B,D3,F)
  tp3.obj(C,D2,E,G)
  lib.lib(E)

- When the -start=A,B,E/400,C,D*:F:G/8000 option is specified



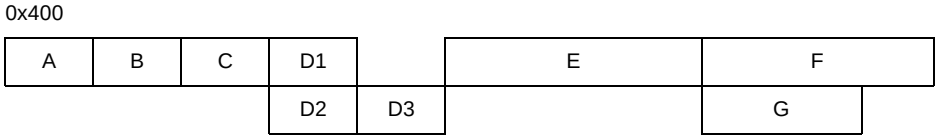
  - Sections C, F, and G delimited by ":" are allocated to the same address.
  - Sections specified with wildcard characters (in this example, the sections whose names start with "D") are allocated in the input order.
  - Objects in the sections having the same name (section E in this example) are allocated in the input order.
  - An input library's sections having the same name (section E in this example) are allocated after the input objects.

- When the -start=A,B,C,D1:D2,D3,E,F:G/400 option is specified



  - The sections that come immediately after ":" (sections A, D2, and G in this example) are selected as the start and allocated to the same address.

- When the -start=A,B,C,(D1:D2,D3),E,(F:G)/400 option is specified



  - When the sections to be allocated to the same address are enclosed by "()", the sections within "()" are allocated to the address immediately after the sections that come before the "()" (sections C and E in this example).
  - The section that comes after the "()" (section E in this example) is allocated after the last of the sections enclosed by "()".

[Remark]

- If the -form={object|relocate|library} option or -strip option is specified, this option will be invalid.
- "()" cannot be nested.
- One or more colons must be described within "()".
  If ":" is not described, "()" cannot be described.
- If "()" is described, ":" cannot be described outside of "()".

-FSymbol

This option outputs external defined symbols to the symbol address file.

[Specification format]

```
-FSymbol=section[,section]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option outputs the external defined symbols in section *section* to a file (symbol address file) in the form of assembler directives.
  The file name is "*output file name*.fsy".
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Section that includes startup function], [Section that outputs external defined symbols to the file] in the [Section] category

[Example of use]

- To output the external defined symbols in sections "sct2" and "sct3" to symbol address file "test.fsy", describe as:

```
>rlink a.obj b.obj -fsymbol=sct2,stc3 -output=test.abs
```

The output example of symbol address file "test.fsy" is shown below.

```
;RENESAS OPTIMIZING LINKER GENERATED FILE xxxx.xx.xx
;fsymbol = sct2,sct3

;SECTION NAME = sct2
    .public _f
_f .equ 0x0
    .public _g
_g .equ 0x16
;SECTION NAME = sct3
    .public _main
_main .equ 0x20
```

[Remark]

- If the -form={object|relocate|library} option or -strip option is specified, this option will be invalid.

-ALIGNED_SECTION

This option changes the number of alignment of the section to 16 bytes.

[Specification format]

```
-ALIGNED_SECTION=section[,section]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option changes the number of alignment of section *section* to 16 bytes.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Section alignment] in the [Section] category

[Example of use]

- To change the number of alignment of section A to 16 bytes, describe as:

```
>rlink a.obj b.obj -aligned_section=A
```

[Remark]

- If the -form={object|relocate|library} option and the -extract or -strip option is specified, this option will be invalid.

Verify specification

The verify specification option is as follows.

- -CPu

-CPu

This option checks the consistency of the address to which the section is allocated.

[Specification format]

```
-CPu=type=address1-address2[,type=address1-address2]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option checks the consistency of the address to which the section is allocated.
  An error will be output if the section allocation address for memory type *type* does not fit in the specified address range.
- The items that can be specified as *type* are shown below.
  An error will occur if any other item is specified.

ROm	Allocates the section to a ROM area.
RAm	Allocates the section to a RAM area.
FIX	Allocates the section to a fixed-address area (e.g. I/O area). If the address range overlaps with ROM or RAM, the setting for FIX is valid.

- Specify the start address and end address of the address range to check for consistency in hexadecimal as *address1* and *address2*.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Check section larger than specified range of address], [Address range of memory type] in the [Verify] category

[Example of use]

- The result is normal when section .text and section .bss are respectively allocated within the ranges from 0x100 to 0x1FF and from 0x200 to 0x2FF.
  If they are not allocated within the ranges, an error will be output.

```
>rlink a.obj b.obj -start=.text/100,.bss/200 -cpu=ROM=100-1FF,RAM=200-2FF
```

[Remark]

- If the -form={object|relocate|library} option or -strip option is specified, this option will be invalid.

Subcommand file specification

The subcommand file specification option is as follows.

- -SUbcommand

-SUbcommand

This option specifies options with a subcommand file.

[Specification format]

```
-SUbcommand=file
```

- Interpretation when omitted
  None

[Detailed description]

- This option specifies options with subcommand file *file*.
- Option contents specified with a subcommand file are expanded to the location at which this option is specified on the command line and are executed.
- See "B.4.2 Subcommand file usage" for details about a subcommand file.

[Example of use]

- Create subcommand file "test.sub" with the following content.

```
input file2.obj file3.obj       ; This is a comment.
library lib1.lib, &             ; This is a line continued.
lib2.lib
```

To specify subcommand file test.sub, describe as:

```
>rlink file1.obj -subcommand=test.sub file4.obj
```

The command line is expanded as follows, and the file input order is: file1.obj, file2.obj, file3.obj, file4.obj.

```
>rlink file1.obj file2.obj file3.obj -library=lib1.lib,lib2.lib file4.obj
```

Other

Other options are as follows.

- -S9
- -STACk
- -COmpress
- -NOCOmpress
- -MEMory
- -REName
- -DELete
- -REPlace
- -EXTract
- -STRip
- -CHange_message
- -Hide
- -Total_size
- -LOgo
- -NOLOgo
- -END
- -EXIt

-S9

This option outputs the S9 record at the end.

[Specification format]

```
-S9
```

- Interpretation when omitted
  None

[Detailed description]

- This option outputs the S9 record at the end even if the entry address exceeds 0x10000.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Hex Output Options] tab, [Output S9 record at the end] in the [Hex Format] category

[Example of use]

- To output the S9 record at the end even if the entry address exceeds 0x10000, describe as:

```
>rlink a.obj b.obj -form=stype -s9
```

[Remark]

- If the -form=stype option is not specified, this option will be invalid.

-STACk

This option outputs the stack information file.

[Specification format]

```
-STACk
```

- Interpretation when omitted
  None

[Detailed description]

- This option outputs the stack information file.
- The file name is "*output-file-name*.sni".
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Output stack information file] in the [Others] category

[Example of use]

- To output stack information file "c.sni", describe as:

```
>rlink a.obj b.obj -stack
```

[Remark]

- If the -form={object|relocate|library} option or -strip option is specified, this option will be invalid.

-COmpress

This option compresses the debug information.

[Specification format]

```
-COmpress
```

- Interpretation when omitted
  The debug information is not compressed (It is the same result as when the -nocompress option is specified).

[Detailed description]

- This option compresses the debug information.
- By compressing the debug information, the loading speed of the debugger is improved.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Compress debug information] in the [Debug Information] category

[Example of use]

- To compress the debug information, describe as:

```
>rlink a.obj b.obj -compress
```

[Remark]

- If the -form={object|relocate|library|hexadecimal|stype|binary} option or -strip option is specified, this option will be invalid.

-NOCOmpress

This option does not compress the debug information.

[Specification format]

```
-NOCOmpress
```

- Interpretation when omitted
  The debug information is not compressed.

[Detailed description]

- This option does not compress the debug information.
- Link time when specifying this option is shorter than when the -compress option is specified.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Compress debug information] in the [Debug Information] category

[Example of use]

- Not to compress the debug information, describe as:

```
>rlink a.obj b.obj -nocompress
```

-MEMory

This option specifies the memory size occupied during linking.

[Specification format]

```
-MEMory=[occupancy]
```

- Interpretation when omitted
  The processing is the same as usual (It is the same result as when the -memory=high option is specified).

[Detailed description]

- This option specifies memory size *occupancy* occupied during linking.
- The items that can be specified as *occupancy* are shown below.

High	The processing is the same as usual.
Low	The optimizing linker loads the information necessary for linking in smaller units to reduce the memory occupancy. This increases the frequency of file access. As a result, processing will be slower than when "High" is specified if the memory used is not larger than implementation memory.

- If *occupancy* is omitted, it is assumed that "High" has been specified.
- Specify "Low" as *occupancy* if processing is slow because a large project is linked and the memory size occupied by the optimizing linker exceeds the available memory in the machine used.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Reduce memory occupancy] in the [Others] category
  - From the [Create Library Options] tab, [Reduce memory occupancy] in the [Others] category

[Example of use]

- To reduce the memory occupancy, describe as:

```
>rlink a.obj b.obj -memory=low
```

[Remark]

- In the following cases, the specification of the -memory=low option will be invalid.
  - When the -form={absolute|hexadecimal|stype|binary} option and following options are specified at the same time
    - Any of the -compress, -delete, -rename, -map, or -stack options
    - The -list and -show[={reference|xreference}] options are specified at the same time.
  - When the -form=library option and following options are specified at the same time
    - Any of the -delete, -rename, -extract, -hide, or -replace options
  - When the -form={object|relocate} option and following options are specified at the same time
    - -extract option

  Some combinations of this option and the input or output file format are invalid.
  See "Table B-5. Relations Between Output Formats And Input Files Or Other Options" for details.

-REName

This option changes an external symbol name or a section name.

[Specification format]

```
-REName={[file](name1=name2[,name1=name2]...)|[module](name1=name2[,name1=name2]...)}
[[,{[file](name1=name2[,name1=name2]...)|[module](name1=name2[,name1=name2]...)}]...]
```

- Interpretation when omitted
  None

[Detailed description]

- This option changes an external symbol name or a section name.
- Specify the symbol name or section name to be changed as *name1*. Specify the symbol name or section name after changing as *name2*.
- Symbol names or section names in specific file *file* or a module in specific library *module* can be changed.
- When a C variable name is specified, add "_" at the head of the definition name in the program.
- When a function name is changed, the operation is not guaranteed.
- If the specified name matches both section and symbol names, the symbol name is changed.
- If there are two or more files or modules with the same name, the priority depends on the input order.

[Example of use]

- To change symbol name "_sym1" to "data", describe as:

```
>rlink a.obj b.obj -rename=(_sym1=data)
```

- To change section ".text" in library module "lib1" to "data" to section "P", describe as:

```
>rlink -library=lib.lib -rename=lib1(.text=P) -form=library
```

[Remark]

- If this option is specified together with the -extract option or -strip option, this option will be invalid.
- When the -form=absolute option is specified, the section name of the input library cannot be changed.
- Operation is not guaranteed if this option is used in combination with compile option -Xmerge_files.

-DELete

This option deletes an external symbol name or a library module.

[Specification format]

```
-DELete={[file](symbol[,symbol]...)|module}[,{[file](symbol[,symbol]...)|module}]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option deletes external symbol name *symbol* or library module *module*.
- Symbol names or modules in specific file *file* can be deleted.
- When a C variable name or C function name is specified, add "_" at the head of the definition name in the program.
- If there are two or more files with the same name, the priority depends on the input order.
- When a symbol is deleted using this option, the object is not deleted but the attribute is changed to the internal symbol.

[Example of use]

- To delete symbol name "_sym1" in all the files, describe as:

```
>rlink a.obj -delete=(_sym1)
```

- To delete symbol name "_sym2" in b.obj, describe as:

```
>rlink a.obj b.obj -delete=b.obj(_sym2)
```

[Remark]

- If this option is specified together with the -extract option or -strip option, this option will be invalid.
- When the -form=library option is specified, library modules can be deleted.
- When the -form={absolute|relocate|hexadecimal|stype|binary} option is specified, external symbols can be deleted.
- Operation is not guaranteed if this option is used in combination with compile option -Xmerge_files.

-REPlace

This option replaces library modules.

[Specification format]

```
-REPlace=file[(module[,module]...)][,file[(module[,module]...)]]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option replaces specified file *file* or library module *module* with the module having the same name in the library file specified by the -library option.

[Example of use]

- To replace file1.obj with module "file1" in library file lib1.lib, describe as:

```
>rlink -library=lib1.lib -replace=file1.obj -form=library
```

- To replace module "mdl1" with module "mdl1" in library file lib1.lib, describe as:

```
>rlink -library=lib1.lib,lib2.lib -replace=lib1.lib(mdl1) -form=library
```

[Remark]

- If the -form={object|relocate|absolute|hexadecimal|stype|binary} option and the -extract or -strip option is specified, this option will be invalid.
- Operation is not guaranteed if this option is used in combination with compile option -Xmerge_files.

-EXTract

This option extracts library modules.

[Specification format]

```
-EXTract=module[,module]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option extracts library module *module* from the library file specified by the -library option.

[Example of use]

- To extract module "file1" from library file "lib.lib" and output it to a file with the object file output format, describe as:

```
>rlink -library=lib1.lib -extract=file1 -form=obj
```

[Remark]

- If the -form={absolute|hexadecimal|stype|binary} option and the -strip option is specified, this option will be invalid.
- When the -form=library option is specified, library modules can be deleted.
- When the -form={absolute|relocate|hexadecimal|stype|binary} option is specified, external symbols can be deleted.

-STRip

This option deletes debug information in the load module file or library file.

[Specification format]

```
-STRip
```

- Interpretation when omitted
  None

[Detailed description]

- This option deletes debug information in the load module file or library file.
- The files before debug information is deleted are backed up in file "*file-name*.abk".
- If this option is specified, one input file should correspond to one output file.

[Example of use]

- To delete debug information of file1.abs, file2.abs, and file3.abs, and output these to file1.abs, file2.abs, and file3.abs, respectively, describe as:
  The files before debug information is deleted are backed up in file1.abk, file2.abk, and file3.abk.
  <Command line>

```
>rlink -subcommand=sub.txt
```

<Subcommand file "sub.txt">

```
-input=file1.abs file2.abs file3.abs
-strip
```

[Remark]

- If the -form={object|relocate|hexadecimal|stype|binary} option is specified, this option will be invalid.

-CHange_message

This option changes the type of information, warning, and error messages.

[Specification format]

```
-CHange_message=level[=num[-num][,num[-num]]...][,level[=num[-num][,num[-num]]...]]...
```

- Interpretation when omitted
  None

[Detailed description]

- This option changes type *level* of information, warning, and error messages.
- The execution continuation or abort at the message output.
- The items that can be specified as *level* are shown below.

Information	Information
Warning	Warning
Error	Error

- If message number *num* is specified, the type of the message with the specified number is changed.
  Also, a range of message numbers can be specified using a hyphen (-).
- Specify the 4-digit number that is output after the component number (05) and the phase of occurrence (6) as *num* (for example, specify 2310 for message number E0562310).
- If the specification of a message number is omitted, the types of all messages are changed to the specified one.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Change warning message to information message], [Number of warning message], [Change information message to warning message], [Number of information message], [Change information and warning message to error message], [Number of information and warning message] in the [Message] category
  - From the [Create Library Options] tab, [Change warning message to information message], [Number of warning message], [Change information message to warning message], [Number of information message], [Change information and warning message to error message], [Number of information and warning message] in the [Message] category

[Example of use]

- To change "E0561310" to a warning and continue the execution at the "E0561310" output, describe as:

```
>rlink a.obj b.obj -change_message=warning=1310
```

- To change all information and warning messages to error messages, describe as:
  If a message is output, the execution will abort.

```
>rlink a.obj b.obj -change_message=error
```

-Hide

This option deletes local symbol name information from the output file.

[Specification format]

```
-Hide
```

- Interpretation when omitted
  None

[Detailed description]

- This option deletes local symbol name information from the output file.
- Since the name information regarding local symbols is deleted, local symbol names cannot be checked even if the file is opened with a binary editor.
  This option does not affect the operation of the generated file.
- Use this option to keep the local symbol names secret.
- The following types of symbols are hidden:
  - C source: Variable or function names specified with the static qualifiers
  - C source: Label names for the goto statements
  - Assembly source: Symbol names of which external definition (reference) symbols are not declared

  The entry function name is not hidden.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Delete local symbol name information] in the [Debug Information] category
  - From the [Create Library Options] tab, [Delete local symbol name information] in the [Debug Information] category

[Example of use]

- To delete local symbol name information from the output file, describe as:

```
>rlink a.obj b.obj -hide
```

The C source example in which this option is valid is shown below.

```
int g1;
int g2=1;
const int g3=3;
static int s1;          //<--- The static variable name will be hidden.
static int s2=1;        //<--- The static variable name will be hidden.
static const int s3=2;  //<--- The static variable name will be hidden.

static int sub1()       //<--- The static variable name will be hidden.
{
    static int s1;      //<--- The static variable name will be hidden.
    int l1;

    s1 = l1; l1 = s1;
    return(l1);
}

int main()
{
    sub1();
    if (g1==1)
        goto L1;
    g2=2;
L1:                     //<--- The label name of the goto statement will be hidden.
    return(0);
}
```

[Remark]

- This option is valid only when the -form={absolute|relocate|library} option is specified.
- When this option is specified with the external variable access optimization, do not specify it for the first linking, and specify it only for the second linking.
- The symbol names in the debug information are not deleted by this option.

-Total_size

This option displays the total size of sections after the linking to the standard error output.

[Specification format]

```
-Total_size
```

- Interpretation when omitted
  None

[Detailed description]

- This option displays the total size of sections after the linking to the standard error output.
- The sections are categorized as follows, with the overall size of each being displayed.
  - Executable program sections
  - Non-program sections allocated to the ROM area
  - Sections allocated to the RAM area
- This option makes it easy to see the total sizes of sections allocated to the ROM and RAM areas.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Display total size of sections] in the [Others] category
  - From the [Create Library Options] tab, [Display total size of sections] in the [Others] category

[Example of use]

- To display the total size of sections after the linking to the standard error output, describe as:

```
>rlink a.obj b.obj -total_size
```

[Remark]

- The -show=total_size option must be specified in order to output the total sizes to the link map file.
- When the ROM-support function (-rom option) has been specified for a section, the section will be used by both the source (ROM) and destination (RAM) of the transfer. The sizes of sections of this type will be added to the total sizes of sections in both ROM and RAM.

-LOgo

This option outputs the copyright notice.

[Specification format]

```
-LOgo
```

- Interpretation when omitted
  This option outputs the copyright notice.

[Detailed description]

- This option outputs the copyright notice.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Display copyright information] in the [Others] category
  - From the [Create Library Options] tab, [Display copyright information] in the [Others] category

[Example of use]

- To output the copyright notice, describe as:

```
>rlink a.obj b.obj -logo
```

-NOLOgo

This option suppresses the output of the copyright notice.

[Specification format]

```
-NOLOgo
```

- Interpretation when omitted
  The copyright notice is output (It is the same result as when the -logo option is specified).

[Detailed description]

- This option suppresses the output of the copyright notice.
- This option is equivalent to the following property in CubeSuite+.
  - From the [Link Options] tab, [Display copyright information] in the [Others] category
  - From the [Create Library Options] tab, [Display copyright information] in the [Others] category

[Example of use]

- To suppress the output of the copyright notice, describe as:

```
>rlink a.obj b.obj -nologo
```

-END

This option executes option strings specified before this option.

[Specification format]

```
-END
```

- Interpretation when omitted
  None

[Detailed description]

- This option executes option strings specified before this option.
  After link processing is terminated, option strings specified before this option are input and link processing is continued.

[Caution]

- This option can be used only in a subcommand file.

[Example of use]

- Create subcommand file "test.sub" with the following content.

```
input=a.obj,b.obj                ;(1)
start=P,C,D/100,B/8000           ;(2)
output=a.abs                     ;(3)
end
input=a.abs                      ;(4)
form=stype                       ;(5)
output=a.mot                     ;(6)
```

To specify subcommand file test.sub, describe as:

```
>rlink -subcommand=test.sub
```

Processing from (1) to (3) are executed and a.abs is output.
Then processing from (4) to (6) are executed and a.mot is output.

-EXIt

This option specifies the end of option specifications.

[Specification format]

```
-EXIt
```

- Interpretation when omitted
  None

[Detailed description]

- This option specifies the end of option specifications.

[Caution]

- This option can be used only in a subcommand file.

[Example of use]

- Create subcommand file "test.sub" with the following content.

```
input=a.obj,b.obj               ;(1)
start=P,C,D/100,B/8000          ;(2)
output=a.abs                    ;(3)
exit
```

To specify subcommand file test.sub, describe as:

```
>rlink -subcommand=test.sub -nodebug
```

Processing from (1) to (3) are executed and a.abs is output.
The -nodebug option specified on the command line after this option is executed is invalid.

B.6 Specifying Multiple Options

This section describes the operation when two or more options are specified for the ccrh command at the same time.

B.6.1 Priority

The following options disable other options.

-V/-h	All options will be invalid. At this time, a warning will not be output.
-P	Since execution is terminated at preprocess processing, options related to the following processing will be invalid. At this time, a warning will be output.
-S	Since execution is terminated at compile processing, options related to the following processing will be invalid. At this time, a warning will be output.
-c	Since execution is terminated at assemble processing, options related to the following processing will be invalid. At this time, a warning will be output.
-Xcpu=g3k	The -Xfloat option will be invalid. At this time, a warning will be output.

If options are specified by the following combinations, the option specified last will be valid with outputting a warning.

- -P, -S, -c
- -D, -U (When their symbol names are same.)
- -Onothing, -Odefault, -Osize, -Ospeed

Depending on the order of specified options, the following options will be invalid.

- -O*item*[Note] that is specified before -Onothing, -Odefault, -Osize, or -Ospeed

Note -O*item*: -Ounroll, -Oinline, -Oinline_size, -Odelete_static_func, -Opipeline, -Otail_call, -Omap, -Osmap

B.6.2 Incompatible features

If options are specified by the following combinations, an error will occur.

- -Omap, -Xep=fix
- -Osmap, -Xep=fix

B.6.3 Dependencies

The behavior of the following options varies depending on what other options are specified.

-Xpreprocess	This option will be invalid if the -P option is not specified at the same time. At this time, a warning will not be output.
-o	If the -P, -S, or -c option is specified at the same time, then the generated file types will be a preprocessed file, assembly source file, or object file.
-g	If the -O option is specified at the same time, debug information may not be correct.
-Oinline	If this option is specified at the same time with the -Xmerge_files option, inline expansion may be performed between files.

B.6.4 Relationship with #pragma directives

The behavior of the following options varies depending on the relationship with #pragma directives.

- When the -Xep=callee or -Xep option is not specified

If the following attribute strings are specified at the same time in a #pragma section, an error will occur.

edata, edata23, tdata, tdata4, tdata5, tdata7, tdata8, sdata, sdata23, ep_auto, ep_disp4, ep_disp5, ep_disp7, ep_disp8, ep_disp16, ep_disp23, gp_disp16, gp_disp23

APPENDIX C INDEX

Symbols

@ (Assemble option) ... 420

@ (Compile option) ... 399

A

Active project ... 66

Add a build mode ... 68

Adding a file ... 14

Add Excluding File dialog box ... 309

Add Existing File dialog box ... 299

Add File dialog box ... 260

Add Folder and File dialog box ... 262

Add Overlay dialog box ... 280

Add Section dialog box ... 280

-ALIGNED_SECTION (Link option) ... 455

Assemble list file ... 83

- Assemble list ... 84
- Command line information ... 85
- Section list ... 85
- Structure of the assemble list ... 83

B

Batch build ... 73, 77

Batch Build dialog box ... 291

-Binary (Link option) ... 427

Browse For Folder dialog box ... 303

Build ... 73, 75

Build mode ... 68, 69

Build Mode Settings dialog box ... 289

Build tool version ... 13

-BYte_count (Link option) ... 444

C

-c (Compile option) ... 346

category ... 19

Change the build mode ... 69

Change the output file name ... 24

-CHange_message (Link option) ... 468

Character String Input dialog box ... 264

Clean ... 81

-COmpress (Link option) ... 460

-CPu (Link option) ... 456

D

-D (Compile option) ... 347, 410

-DEBug (Link option) ... 433

-DEFine (Link option) ... 428

-DELete (Link option) ... 464

Delete a build mode ... 71

E

Editor panel ... 256

-END (Link option) ... 474

-ENTry (Link option) ... 429

-EXIt (Link option) ... 475

-EXTract (Link option) ... 466

F

File dependencies ... 20

File display order ... 20

-FOrm (Link option) ... 430

-FSymbol (Link option) ... 454

G

-g (Assemble option) ... 407

-g (Compile option) ... 341

[General - Build/Debug] category ... 296

H

-h (Assemble option) ... 403

-h (Compile option) ... 335

-Hide (Link option) ... 469

I

-I (Assemble option) ... 412

-I (Compile option) ... 349

Import Build Options dialog box ... 301

-Input (Link option) ... 423
Intel HEX file ... 100
Data record ... 103
End record ... 104
Expanded address record ... 103
Expanded linear address record ... 102
Start address record ... 102
Start linear address record ... 102
Structure of the Intel HEX file ... 100

L

-LIBrary (Link option) ... 425
Library list file ... 96
Error information ... 97
Library information ... 97
Module, section, and symbol information within the library ... 98
Option information ... 96
Structure of the library list ... 96
Link map file ... 86
Cross reference information ... 90
Error information ... 87
Link map information ... 88
Option information ... 87
Structure of link map ... 86
Symbol information ... 89
Total section size ... 88
Link Map File (When Objects Are Combined) ... 92
Address overlap information ... 95
Combined address information ... 94
Entry information ... 94
Error information ... 94
Header information ... 93
Option information ... 93
Structure of link map ... 92
Link Order dialog box ... 287
-LISt (Link option) ... 449
-LOgo (Link option) ... 472

M

Main window ... 112
-MAp (Link option) ... 439
-MEMory (Link option) ... 462
-Message (Link option) ... 441
Modify Section dialog box ... 280
Motorola S-record file ... 105
S0 record ... 106
S1 record ... 107
S2 record ... 107
S3 record ... 107
S7 record ... 108
S8 record ... 108
S9 record ... 109
Structure of the Motorola S-record file ... 105
-MSg_unused (Link option) ... 443

N

-NOCOmpress (Link option) ... 461
-NODEBug (Link option) ... 434
-NOLOgo (Link option) ... 473
-NOMessage (Link option) ... 442

O

-o (Assemble option) ... 404
-O (Compile option) ... 365
-o (Compile option) ... 336
Open with Program dialog box ... 315
Option dialog box ... 294
[General - Build/Debug] category ... 296
-OUtput (Link option) ... 437
Output an assemble list ... 27
Output library information ... 28
Output map information ... 27
Output panel ... 257
-OVERRUN_FETCH (Link option) ... 446

P

-P (Compile option) ... 344
-PADDING (Link option) ... 445
Path Edit dialog box ... 269
Progress Status dialog box ... 293
Project Tree panel ... 116
Property panel ... 129

[Assemble Options] tab ... 175
[Build Settings] tab ... 217
[Category Information] tab ... 254
[Common Options] tab ... 132
[Compile Options] tab ... 153
[Create Library Options] tab ... 205
[File Information] tab ... 252
[Hex Output Options] tab ... 199
[Individual Assemble Options] tab ... 243
[Individual Compile Options] tab ... 220
[Link Options] tab ... 182

R

Rapid build ... 73, 76
Rebuild ... 73, 75
-RECord (Link option) ... 435
-RENname (Link option) ... 463
-REPlace (Link option) ... 465
-ROm (Link option) ... 436
Run a build ... 73

S

-S (Compile option) ... 345
-S9 (Link option) ... 458
Save As dialog box ... 313
Save Settings dialog box ... 285
Section Address dialog box ... 282
Section Settings dialog box ... 277
Select Export File dialog box ... 319
Select Import File dialog box ... 317
Set assemble options ... 35
Set compile options ... 30
Set create library options ... 55
Set hex output options ... 50
Set link options ... 39
-SHow (Link option) ... 450
Simultaneous build ... 78
-SPace (Link option) ... 440
Specify Far Jump File dialog box ... 305
Specify MISRA-C:2004 Rule File dialog box ... 307
Specify Rule Number dialog box ... 275
Specify Using Library File dialog box ... 311
-STACk (Link option) ... 459
-STARt (Link option) ... 452
-STRip (Link option) ... 467
-SUbcommand (Link option) ... 457
System Include Path Order dialog box ... 273

T

Tag jump ... 258
Text Edit dialog box ... 266
-Total_size (Link option) ... 471

U

-U (Assemble option) ... 411
-U (Compile option) ... 348
Unassigned Section dialog box ... 283

V

-V (Assemble option) ... 402
-V (Compile option) ... 334

X

-Xalias (Compile option) ... 373
-Xansi (Compile option) ... 354
-Xasm_far_jump (Assemble option) ... 417
-Xasm_option (Compile option) ... 397
-Xasm_path (Compile option) ... 339, 340
-Xbit_order (Compile option) ... 378
-Xcall_jump (Compile option) ... 385
-Xcharacter_set (Assemble option) ... 413
-Xcharacter_set (Compile option) ... 364
-Xcheck (Compile option) ... 358
-Xcheck_div_ov (Compile option) ... 389
-Xcheck_language_extention (Compile option) ... 363
-Xcommon (Assemble option) ... 408
-Xcommon (Compile option) ... 342
-Xcpu (Compile option) ... 343, 409
-Xcref (Compile option) ... 394
-Xdiv (Compile option) ... 388
-Xenum_type (Compile option) ... 356
-Xep (Assemble option) ... 416
-Xep (Compile option) ... 383

-Xerror_file (Assemble option) ... 418
-Xerror_file (Compile option) ... 395
-Xfar_jump (Compile option) ... 386
-Xfloat (Compile option) ... 384
-Xignore_files_misra (Compile option) ... 362
-Xinline_strcpy (Compile option) ... 371
-Xintermodule (Compile option) ... 369
-Xlk_option (Compile option) ... 398
-Xmerge_files (Compile option) ... 374
-Xmerge_string (Compile option) ... 372
-Xmisra2004 (Compile option) ... 360
-Xmulti_level (Compile option) ... 392
-Xno_warning (Assemble option) ... 419
-Xno_warning (Compile option) ... 396
-Xobj_path (Assemble option) ... 405
-Xobj_path (Compile option) ... 338
-Xpack (Compile option) ... 376
-Xpass_source (Compile option) ... 379
-Xpatch (Compile option) ... 393
-Xpreinclude (Compile option) ... 351
-Xpreprocess (Compile option) ... 352
-Xprn_path (Assemble option) ... 406
-Xreg_mode (Assemble option) ... 414
-Xreg_mode (Compile option) ... 381
-Xreserve_r2 (Assemble option) ... 415
-Xreserve_r2 (Compile option) ... 382
-Xswitch (Compile option) ... 380
-Xunordered_cmpf (Compile option) ... 391
-Xuse_fmaf (Compile option) ... 390
-Xvolatile (Compile option) ... 357
-Xwhole_program (Compile option) ... 375

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Apr 01, 2013	-	First Edition issued

CubeSuite+ V2.01.00 User's Manual:
RH850 Build

Publication Date: Rev.1.00 Apr 01, 2013

Published by: Renesas Electronics Corporation

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

© 2013 Renesas Electronics Corporation and Renesas Solutions Corp.

Colophon 1.3

CubeSuite+ V2.01.00

Renesas Electronics Corporation

R20UT2585EJ0100